PLANNING, ESTIMATION AND CONTROL FOR MOBILE ROBOT LOCALIZATION WITH APPLICATION TO LONG-TERM AUTONOMY

A Dissertation

by

SAURAV AGARWAL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Suman Chakravorty
Committee Members,	John Valasek
	Nancy Amato
	Dylan Shell
	Dezhen Song
Head of Department,	Rodney Bowersox

May 2018

Major Subject: Aerospace Engineering

ABSTRACT

There may arise two kinds of challenges in the problem of mobile robot localization; (i) a robot may have an a priori map of its environment, in which case the localization problem boils down to estimating the robot pose relative to a global frame or (ii) no a priori map information is given, in which case a robot may have to estimate a model of its environment and localize within it. In the case of a known map, simultaneous planning while localizing is a crucial ability for operating under uncertainty. We first address this problem by designing a method to dynamically replan while the localization uncertainty or environment map is updated. Extensive simulations are conducted to compare the proposed method with the performance of FIRM (Feedback-based Information RoadMap). However, a shortcoming of this method is its reliance on a Gaussian assumption for the Probability Density Function (pdf) on the robot state. This assumption may be violated during autonomous operation when a robot visits parts of the environment which appear similar to others. Such situations lead to ambiguity in data association between what is seen and the robot's map leading to a non-Gaussian pdf on the robot state. We address this challenge by developing a motion planning method to resolve situations where ambiguous data associations result in a multimodal hypothesis on the robot state. A Receding Horizon approach is developed, to plan actions that sequentially disambiguate a multimodal belief to achieve tight localization on the correct pose in finite time. In our method, disambiguation is achieved through active data associations by picking target states in the map which allow distinctive information to be observed for each belief mode and creating local feedback controllers to visit the targets. Experiments are conducted for a kidnapped physical ground robot operating in an artificial

maze-like environment.

The hardest challenge arises when no a priori information is present. In long-term tasks where a robot must drive for long durations before closing loops, our goal is to minimize the localization error growth rate such that; (i) accurate data associations can be made for loop closure, or (ii) in cases where loop closure is not possible, the localization error stays limited within some desired bounds. We analyze this problem and show that accurate heading estimation is key to limiting localization error drift. We make three contributions in this domain. First we present a method for accurate long-term localization using absolute orientation measurements and analyze the underlying structure of the SLAM problem and how it is affected by unbiased heading measurements. We show that consistent estimates over a 100km trajectory are possible and that the error growth rate can be controlled with active data acquisition. Then we study the more general problem when orientation measurements may not be present and develop a SLAM technique to separate orientation and position estimation. We show that our method's accuracy degrades gracefully compared to the standard non-linear optimization based SLAM approach and avoids catastrophic failures which may occur due a bad initial guess in non-linear optimization. Finally we take our understanding of orientation sensing into the physical world and demonstrate a 2D SLAM technique that leverages absolute orientation sensing based on naturally occurring structural cues. We demonstrate our method using both high-fidelity simulations and a real-world experiment in a $66,000$ square foot warehouse. Empirical studies show that maps generated by our approach never suffer catastrophic failure, whereas existing scan matching based SLAM methods fail $\approx 50\%$ of the time.

DEDICATION

For my wife Aishwarya, my parents Mukesh and Suneeta Agarwal and my brother Raghav Agarwal. You are the reason I keep going.

ACKNOWLEDGEMENTS

I would like to thank my Ph.D. adviser Prof. Suman Chakravorty for being an inspiration, a friend and a mentor. It was through his constant nudging and cajoling that I was able to discover some of the true underlying problems in autonomous navigation. Suman has taught me to constantly question my assumptions and think deepely about every problem before jumping in to solve it. His advice to not let existing literature corrupt my thought process is something I hold very dear to my heart. I am also grateful to my committee members for being an integral part of this journey. Your support, feedback and criticism has helped me become a better researcher. I want to thank you for dedicating some of your valuable time to my journey.

I am extremely grateful to the team at the Technology Commercialization office, especially Ismail Sheikh who has been critical in helping us patent some of the contributions that are developed in this work. I am forever grateful for your friendship and advice. Many different people at Texas A&M University and the broader ecosystem have played an important role in helping me transition into an entrepreneur. I want to thank Nick Spiller, Prof. Richard Lester, Blake Petty, Don Lewis, Jose Quintana and last but not the least James Lancaster.

I want to thank my friends and room mates Purshottam Bhangui, Vikram Bhargava, Saptaparni Kumar and Swati Singh for being by my side through thick and thin. Without the fun discussions, constant joking around and banter of the office, it would have been hard to stay motivated for the last almost 5 years. I want to thank Bharat and Raman for being my go to movie buddies. I've learned a great many things from my lab mates Amirhossein and Mohammad, often we were the only ones

who could understand each other's graduate school dilemmas. I want to sincerely thank Matthew Skolaut and Karthikeya Sharma for helping me and working with me on the hardware and software that came out of this research.

I must dedicate a separate paragraph for Dr. Ali-akbar Agha-mohammadi. You are and always will be a role model. I am extremely thankful to you for teaching me the ropes of graduate school. From teaching me motion planning under uncertainty to how to take ideas from thoughts to equations to code, your words of wisdom are forever engraved in my mind. Your advice to constantly keep writing my thoughts has helped me stay on top of my work. Without you, the wonderful experience of industrial research at Qualcomm, San Diego would not have been possible. I hope we get the opportunity to collaborate on many more amazing projects in the future.

Last but most important I am grateful to my family. My parents and brother for being the rock solid foundation on which I can always stand confidently and shoot for my dreams. Yet I know that when I take a giant leap and fall, you will always be there to pick me back up. I want to thank my wife Aishwarya, you lost so much in such little time, yet you managed to get up and stand beside me. I can never repay or thank you enough. I draw my strength and courage from you.

CONTRIBUTORS & FUNDING SOURCES

I would like to thank my Ph.D. adviser Dr. Suman Chakravorty and my Committee members Dr. Dylan Shell, Dr. Dezhen Song, Dr. John Valasek and Dr. Nancy Amato in mentoring and advising me through the Ph.D. journey. I would also like to thank lab members and my collaborators Dr. Ali-akbar Agha-mohammadi, Amirhosein Tamjidi, Karthikeya Sharma and Vikram Shree for their inputs to this work.

This work was made possible in part by the National Science Foundation under Grant Numbers NSF-RI-1217991 and NSF-NRI-1637889.

Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the National Science Foundation.

NOMENCLATURE

t_k	Time at step k
x_k	Robot state at time t_k
$\mathbf{x}_k$	Robot state vector at time t_k
$\hat{x}_k^+$	Robot state estimate at time t_k
$\hat{x}_{k+1}^-$	Predicted robot state prior at time t_{k+1}
u_k	Control input at time t_k
z_k	Sensory observation at time t_k
$\mathbb{X}$	State space
$\mathbb{U}$	Control space
$\mathbb{Z}$	Observation space
$\mathcal{N}(\mu, \Sigma)$	Gaussian random variable with mean μ and covariance Σ
l_i	i-th landmark in map
$\mathbf{l}_i$	Position vector for i-th landmark
$\mathbf{p}_k$	Position vector of robot w.r.t global frame at time t_k
$\boldsymbol{\theta}_k$	Euler angles of robot body axis w.r.t global frame at time t_k

AHRS	Attitude Heading Reference Systems
KF	Kalman Filter
DCM	Direction Cosine Matrix
EKF	Extended Kalman Filter
UKF	Unscented Kalman Filter
SEIF	Sparse Extended Information Filter
PDF	Probability Density Function
SLAM	Simultaneous Localization and Mapping
SLAP	Simultaneous Localization and Planning
SPLAM	Simultaneous Planning Localization and Mapping
POMDP	Partially Observable Markov Decision Process
FIRM	Feedback-based Information RoadMaps
GPS	Global Positioning System
MAP	Maximum a Posteriori
ML	Maximum Likelihood
UAV	Unmanned Aerial Vehicle
MAV	Micro Aerial Vehicle
RMSE	Root Mean Squared Error
MPUU	Motion Planning Under Uncertainty
M3P	MultiModal Motion Planning
GMM	Gaussian Mixture Model
MHT	Multi-Hypothesis Tracking
LFGO	Linear Feature Graph Optimization
R.V.	Random variable

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

In the field of autonomous mobile robotics, our goal is to create systems that when switched on can be left to operate with minimal human intervention. To move about in their environment safely and reliably complete assigned tasks, robots need to tackle three fundamental problems:

1. **Localization**: A robot needs to estimate where it is within its map to be able to decide where it should go next.

2. **Mapping**: If a model of the environment is not known a priori then it must estimate such a model to navigate safely.

3. **Planning**: A robot should be able to determine the best future action using currently available knowledge of its state and map to complete assigned tasks.

If a robot is given a map of its environment, then we require that it should plan the best course of action such that it can stay well localized, and complete tasks without colliding with the environment. This problem is referred to as Simultaneous Localization and Planning (SLAP). If a robot is not given a map a priori then it needs to map and localize within its estimated map, this is called Simultaneous Localization and Mapping (SLAM). Finally, to achieve full autonomy, all three problems need to be tackled at once, i.e., a robot needs to plan for a given task while operating in an environment about which it does not possess a priori information. This problem is referred to as Simultaneous Planning Localization and Mapping (SPLAM).

In this work, we study both the SLAP and SLAM problems from the perspective of long-term autonomy and briefly touch upon SPLAM. In the scope of this work, long-term autonomy refers to tasks where a robot may routinely explore

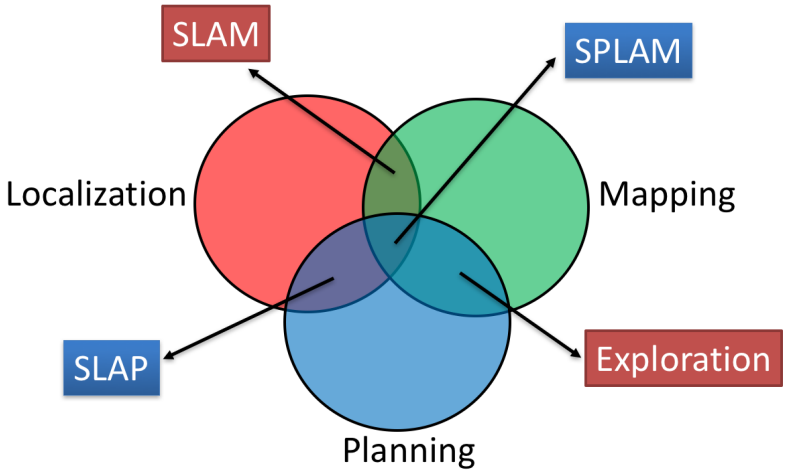


Figure 1.1: A venn diagram depicting the three key problems in mobile robotics.

large unknown regions or persistently operate within a known map where uncertainty necessitates offline plans to be modified online. In the SLAP problem we look at; (i) improving the efficiency of Feedback-based Information RoadMaps (FIRM) through rollout-based methods and (ii) extending motion planning under uncertainty (MPUU) to non-Gaussian domain through active data association for disambiguating multimodal beliefs. In the SLAM problem we study the role of orientation (heading) and propose a novel formulation to achieve highly accurate localization with an unbiased heading sensor (e.g., magnetometer, star tracker, gyrocompass). In a general setting, orientation measurements may not be available, thus we develop a feature-based SLAM approach that separates orientation and position estimation and exploits the separable structure of the SLAM problem. Finally we develop a variant of scan matching based 2D SLAM that leverages orientation sensing using structural cues and demonstrate it on a physical system.

In Sections 1.1 and 1.2, we describe key elements of localization and mapping and discuss commonly known issues. In Section 1.3 we present how MPUU plays an important role in improving localization and mapping quality while increasing the

robustness of task completion.

1.1 Localization in Known Maps

For a known map, the localization problem can be framed as follows, "*given a robot and a map of the environment in which it operates, we require that a robot determine its pose with respect to a global reference frame using available sensors*". Due to the stochastic nature of the problem (noisy sensing and actuation), we modify the above statement to, "*given a robot and a map of the environment in which it operates, we require that a robot estimate a belief about its pose with respect to a global reference frame using available sensors*". Here belief refers to the probability distribution function (pdf) over all possible robot states. When no prior pose information is given to the robot, it is commonly referred to as the global localization problem, sometimes also called the "lost or kidnapped robot problem". Two commonly used approaches to localization are:

1. Feature-based localization: Robot is given the spatial locations of key points or landmarks in the environment. A robot then uses its onboard sensors (e.g. camera, lidar, etc.) to detect and get relative readings to these landmarks and estimate its belief.

2. Grid-based localization: Robot is given an occupancy grid of the world, i.e., a map which grids the world into discrete cells and contains information about the probability of each grid cell being occupied by an obstacle. A robot then uses a range sensor (e.g. sonar, depth camera, lidar etc.) to sense and estimate the probability of being in each grid cell. In this work, we limit ourselves to the study of feature based methods.

In any localization method, challenges arise due to data association between what

is seen and robot's a priori map:

1. Known Data Associations: When a robot can establish a unique data association between every feature it observes to some feature in its map then we call it known data association. This is generally a simpler problem to work with and the belief can be modeled well by a unimodal Gaussian pdf.

2. Unknown Data Associations: Situations may arise where data association between what is observed and the robot's map leads to a multimodal hypothesis on the state, for example a kidnapped robot with no a priori information about its state or a mobile robot operating in a symmetric environment (see Fig. 1.2). In this case, a Gaussian (unimodal) belief assumption may not always be a valid choice. In Chapter 3 we present a motion planning approach for active data association to disambiguate a multimodal belief.

1.2 Simultaneous Localization and Mapping

When a robot is not given a priori knowledge of its environment, it must use its sensory data and actions to concurrently build a map of its environment and localize itself within its stochastic map, this is referred to as Simultaneous Localization and Mapping (SLAM) [136]. Common techniques either use a filtering-based approach to process incoming data sequentially and maintain an up-to-date belief of the robot state and features in the world (e.g., Extended Kalman Filter (EKF) SLAM) or post-process odometery data and measurements in a batch manner using non-linear optimization techniques to estimate the full robot trajectory and map (e.g., pose-graph SLAM). Further, maps can be sparse such that they represent only the locations of landmarks or dense representations can be built which contain information about the world geometry and appearance. An important phenomenon in

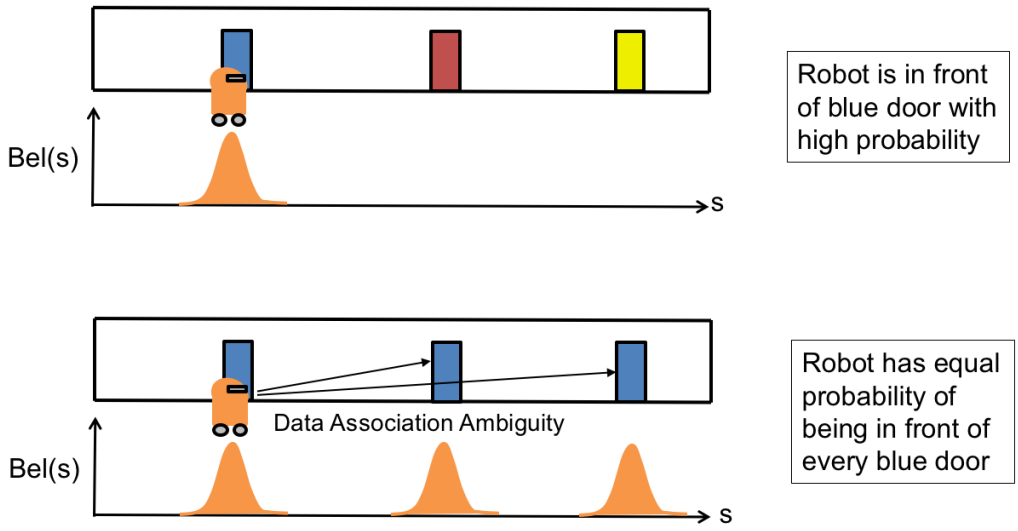


Figure 1.2: A simple depiction showing how symmetry in the environment can lead to ambiguity in data association. In the first case the world has three doors, with unique appearance hence a Gaussian belief assumption works well. In the second case, the world has three identical colored doors, which leads to a multimodal belief.

SLAM is *loop closure* that occurs when a robot re-observes a previously seen part of the map. This often leads to a reduction in the localization and map uncertainty along with biases which built up as a robot previously moved into unknown parts of the map.

Issues with filtering-based methods: Filtering-based methods suffer from quadratic complexity in the number of features tracked, which limits the size of maps that can be built. While methods exist to break the mapping problem into submaps, errors may arise in consistently estimating a global map by fusing local maps. Further, the standard filtering-based formulation is known to be inconsistent as the heading is not observable. This results in biased estimates and the filter becomes overconfident about its knowledge.

Issues with batch methods: In batch methods, we must solve a non-linear optimization problem the size of which can quickly grow due to the high frequency

of incoming observations. Further, non-linear optimization requires a good initial guess to converge and generally, odometery is used to provide this initial guess. However odometery based estimation error grows unbounded as the robot moves. Thus, associating new observations to previously seen sections of the map for loop closure is often difficult with disastrous consequences in map building. A common approach is to build a two part solution with a backend and front end. While the back end solves the non-linear optimization problem, the front end handles data association. Usually the front end operates independently of the backend and does not use prior knowledge of robot pose to improve data association. Thus any data association error made in the front end is propagated to the map building process. Most importantly, any non-linear optimization technique cannot guarantee a globally optimal solution.

1.3 Motion Planning Under Uncertainty

The role of motion planning is to determine actions that drive a robot from a given start state to some goal state. Due to the stochastic nature of real-world robotics, our goal is often to maximize the probability of reaching the goal state. Thus actions must be planned in the space of beliefs over the robot state if the map is known or in the entire space of beliefs over robot state and map. In general, the problem of motion planning under uncertainty belongs to the class of problems known as Partially Observable Markov Decision Processes (POMDP). Motion Planning Under Uncertainty is also referred as *belief space planning* in robotics literature.

SLAP: In a known map, there may be certain parts of the environment which provide richer localization information as compared to others. Thus motion planning often incorporates the uncertainty in robot localization as a cost to be minimized thus leading to information seeking behavior that enhances localization certainty to gain

robustness in the face of noises and perturbations while completing tasks. Figure 1.3 shows an example from [10] which depicts a robot operating in a 2D environment. A traditional deterministic planner would drive a robot through the shortest path (green) but an uncertainty aware planner like FIRM chooses a path (red) that drives the robot closer to landmarks to enhance localization. In Chapter 2 we present two contributions to the problem of SLAP via rollout-based planners for Gaussian belief space and active data association for non-Gausian belief space planning.

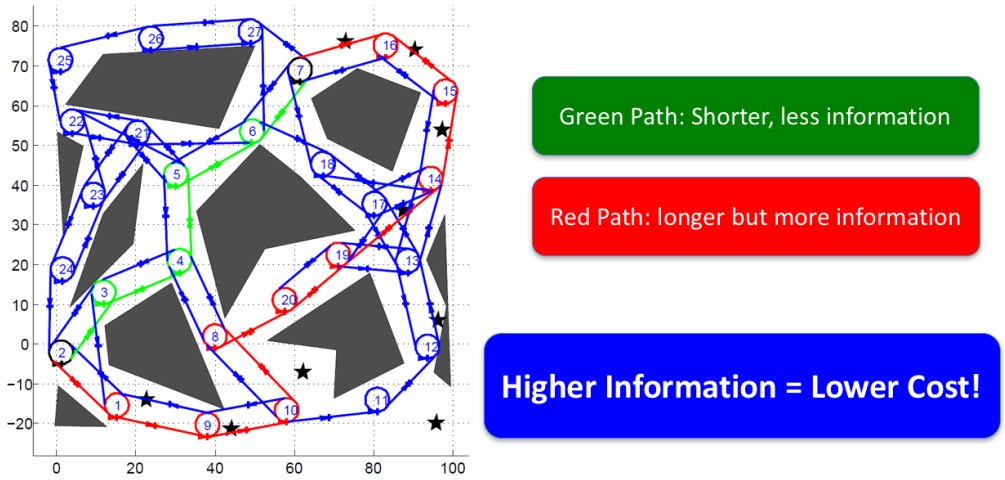


Figure 1.3: SLAP often leads to informative paths as compared to geometrically shortest path [10].

SPLAM: In SLAM, it is well understood that mapping and localization accuracy can be significantly improved by taking deliberate actions, this is referred to in literature as "active" SLAM. In active SLAM, a common approach is to plan actions that achieve a balance between exploration (i.e., visiting previously unseen areas) and exploitation (i.e., revisiting known areas), such that overall map uncertainty is reduced. Both active SLAM and traditional SLAM focus on building accurate models of the world. Currently, these methods are limited to trajectories generally $<$ 1km.

For problems where the distance traveled may routinely exceed these limits (e.g. planetary rovers) it may be impractical to close loops and get rid of biases. In Chapter 5 we propose a mapping formulation to achieve consistent feature localization with an unbiased heading sensor.

1.4 Outline

In Chapter 2 we present a literature survey of prior work in SLAM and look at the problem of SLAP for real-time replanning. In Chapter 3 we present a method for SLAP in non-Gaussian belief spaces. In Chapter 4 we survey prior work in the field of SLAM and discuss the role of orientation in SLAM. In Chapters 5 and 6 we present novel methods for long-term mapping and localization with absolute orientation sensing and without orientation measurements respectively. In Chapter 7 we present a novel method for robust scan matching-based 2D SLAM which leverages absolute orientation and show its benefits in practice. Finally in Chapter 8 we conclude this work and discuss directions for future research.

2. SIMULTANEOUS LOCALIZATION AND PLANNING

The field of SLAP has built up a large body of contributions in a relatively short span of time. Belief space planning methods range from sampling based approaches to optimization-based methods to a combination of both. In Section 2.1 we discuss relevant prior art in the field of SLAP with a focus on sampling-based methods. We pay particular attention to works that study the problem of global localization (Section 2.1.1) and online replanning (Section 2.1.2) as these are most relevant in the context of this dissertation. In Section 2.2 we touch upon a Rollout-based extension of FIRM for real-time Simultaneous Localization and Planning[1].

2.1 Literature Review

Recent work in sampling-based methods for belief space planning have shown promising results. The fundamental goal being to plan actions that minimize uncertainty such that a mobile robot can localize accurately to act safely and reliably. Methods such as [119, 21, 30, 143, 85, 117] provide solutions that depend on the initial belief. In [119], the authors present an information space planning method. First a PRM is constructed where the nodes are the means of beliefs. These beliefs are given a stationary covariance and connected via edges. Paths are chosen which result in the lowest uncertainty at the goal state. In [21], the authors construct a graph in belief space then prune it regularly. They prove that the method converges to the optimal solution as the number of samples tends to infinity. In [30], the authors present a method for making discrete time approximations to continuous time POMDPs and use SARSOP to solve the discrete-time POMDP. SARSOP [86] is a POMDP solver which builds a solution by approximating through sampling, the

[1]Part of this section was reprinted with permission from [9].

reachable region of the belief space by an optimal policy. SARSOP builds a tree in belief space, by sampling an action $a \in A$ and applying it to the belief b, then using the transition function to compute the child of b. A lower and upper bound on the Value function is assumed and program terminates when the gap between the upper and lower bounds converges to some value $\leq \epsilon$. The method of [143] generates multiple nominal paths using RRT (or some other planner), then predicts the a-priori belief behavior over these trajectories and finally picks the best one. The planning objective is to minimize the collision probability but instead of directly computing collision probability this metric is approximated by the number of standard deviations a robot can deviate from its path before it collides. In [16], the authors present a method called Robust Belief Roadmap which is a motion planning algorithm for systems with uncertainty that can reason about noisy intermittent observations. Sensor misdetection is a challenging problem faced by mobile robots in real world applications as different sensors are affected by the environment in various ways that can cause these sensors to not detect landmarks. The authors propose to use the expectation of the maximum eigen value of the covariance matrix as a metric for the uncertainty at the goal and develop an analytical bound on this metric. They assume that the missed detection probability can quantified and that the misdetections are independent over time and between sensors. A computationally tractable recursive method is presented to compute the maximum bound on the metric. Recent developments in [10, 9] extend belief space planning to multi-query settings (cases where multiple planning requests are made sequentially) by creating a belief space variant of a Probabilistic RoadMap (PRM) [76]. We note that all the methods mentioned above rely on a Gaussian belief assumption. Additionally, the aforementioned methods assume that the data associations between observations and information sources (e.g., landmarks) are known and unambiguous.

Another class of methods is the trajectory optimization approach which can be implanted in a Receding Horizon Control (RHC) framework for planning. A widely used approach in RHC-based control is to approximate the stochastic system with a deterministic one by substituting the random variables with their most-likely values [15]. Methods such as [29, 117, 58, 145], assume the most-likely values for the unknown future observations in the planning stage.

Recent work in [115, 116] extends belief space planning to non-Gaussian beliefs where the belief modes are not widely separated. The authors investigate a grasping problem with a multimodal hypothesis on the gripper's state. Their method picks the most-likely hypothesis and a fixed number of samples from the belief distribution, then using an RHC approach, belief space trajectories are found that maximize the observation gap between the most-likely hypothesis and the drawn samples, which helps to accept or reject the most-likely hypothesis. The method in [114] builds upon the work in [115] wherein the author transposes the non-convex trajectory planning problem in belief space to a convex problem. Among other recent works, [121] reduces the computational complexity of planning for a non-Gaussian hypothesis but also assumes distributions without widely separated modes.

2.1.1 Global Localization

To the extent of our knowledge, a limited number of methods approach the problem of recovering global robot pose for a mobile robot with an initial multimodal hypothesis. The analysis in [41] showed that finding the optimal (shortest) plan to re-localize a robot with multiple hypotheses in a deterministic setting (no sensing or motion uncertainty) is NP-hard. At best a greedy localization strategy can be developed whose plan length is upper bounded by a factor of the optimal plan. In the localization strategy of [41] reference points are chosen in the environment at

which observations may lead to disambiguation, the robot is then driven the minimum distance over all active hypothesis-reference point combinations to make a perfect range-scan observation with infinite range (not available in practice). Section 3.3 presents a discussion on differences of [41] from M3P that highlight why [41] may not work well for a physical robot. In a symmetric environment, [107] showed that for a robot equipped with only perfect odometery, no sequence of actions can disambiguate a pair of symmetric configurations. In [50], the authors develop an active localization method in a grid based scheme for a known map. Their planning method considers arbitrary targets in the robot's local coordinate frame as atomic actions (e.g., move 1m right and 4m forward). The optimal candidate action is selected based on the path cost and the expected decrease in entropy at the target. Successful application of the Gaussian mixture model to multi-hypothesis tracking for robot localization was shown in [122, 64, 124]. In [64], the authors present a greedy heuristic-based planning strategy to disambiguate a multimodal hypothesis for a kidnapped robot. The method of [53] plans safe trajectories by picking a point in the vicinity of obstacles to disambiguate the hypothesis.

In Chapter 3 we propose a novel algorithm called Multi-Modal Motion Planner (M3P) to handle global localization. Compared to [50], our target selection methodology is active, i.e., M3P uses the a priori map information to select targets such that by visiting them, belief modes expect to see disambiguating information (e.g., seeing a unique landmark that can immediately confirm or reject a hypothesis, see Fig. 3.2). Compared to [64, 53], we present a planning approach that explicitly reasons about the belief evolution as a result of actions in the planning stage and picks an optimal policy from a set of candidates. Compared to [115, 116, 114], our method is better suited to deal with more severe cases of non-Gaussian belief space planning such as the kidnapped robot scenario. Such scenarios may not be possible

to address using the trajectory optimization based techniques of [114, 115] in their current form, due to the difficulty of generating an initial feasible plan for the widely separated modes in the presence of obstacles (see Fig. 3.1 for an example of widely separated modes).

2.1.2 Online Replanning

The majority of planners designed to solve POMDPs are not equipped with online replanning capabilities. Sampling-based methods (e.g., [119] [144], [21], [85]) and point-based methods (e.g., [113], [86], [12], [30], [123], [134], [109]) are single query methods, thus in case of replanning from a new belief most of the computation needs to be redone, in particular when the planner has to switch the plan from one homotopy class to another. The method proposed in [87] reuses offline computations and demonstrate online replanning in simulation, where it chooses between two heuristics (with adaptive weights) to ascertain whether the planner needs to explore new actions or exploit its knowledge.

From an experimental point of view, a few recent works have focused on applying belief space planning to real-world robots. The authors of [129] present a planner named Expected Shortest-Path Elastic Roadmap (ESPER) that integrates task constraints while generating whole body motion plans with reasoning about uncertainty and demonstrate it in an experiment. The Elastic RoadMap [150] is a feedback motion planning method which generates reactive, task-constrained, whole body motion for manipulation. [68] is an integrated task and motion planner in belief space, utilizing symbolic abstraction, whose performance is demonstrated on a PR2 robot tasked with picking and placing household objects. In [20], the authors develop a motion planning algorithm that is able to react to changes in the environment in real time for executing learned tasks. Using a Hidden Markov Model to represent the learned

task model and taking its cartesian product with an offline generated PRM, the planner finds plans in the spatiotemporal roadmap. During plan execution, replanning is carried out a high frequency during which newly observed constriaints are used to update the roadmap edges and compute updated plans.

FIRM [10] extended Probabilistic Roadmaps [75, 89] to SLAP wherein the feedback policy is the Dynamic Programming (DP) solution over the graph. However, following a FIRM feedback policy requires stabilization at each FIRM node which may make a robot inefficient. We now proceed to present results for simulation studies of rollout-based FIRM and its comparison to the original FIRM methodology along with a local optimization-based method.

2.2 Rollout-based Extension to FIRM

We briefly present the concept of rollout-based belief space planning and discuss simulation results which demonstrate the performance improvements made by using the proposed method. For an in-depth description of the methodology, theoretical analysis and experimental results, we refer the reader to a preprint version of this work [11]. The proposed method supercedes [10], in performance, success probability, and ability to cope with changing environments. It builds upon a FIRM and inherits the desired features of the FIRM framework such as robustness, scalability, and feedback nature of the solution. But, it also eliminates the need for belief node stabilization in the original FIRM method, thus can be viewed as a stabilization-free FIRM. We also present comparisons against a state-of-the-art local optimization-based method.

2.2.1 Methodology

Once a FIRM policy [10] is computed and the robot starts following it, during each step of policy execution, the planner checks local connections to neighboring FIRM

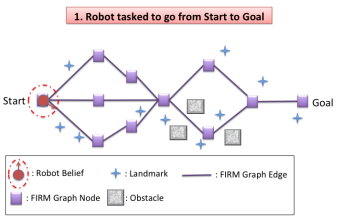
nodes. It computes cost-to-connect to neighboring FIRM nodes through monte carlo simulations. Using knowledge of the FIRM cost-to-go as the base cost, the planner then computes the cost to reach the goal through the rollout connection. Thus, by embedding an online local planning module in the FIRM framework, we achieve a locally optimal tradeoff between stabilization to a node (i.e., exploring the information space to reach the exact belief node) and moving forward towards goal (exploiting the gradient of local cost function), while the global optimality on the graph is still guaranteed by solving dynamic programming. As a result of this optimal tradeoff, interesting behaviors emerge out of the algorithm without encoding any heuristic. For example, consider a case when the desired cost is to "reach a goal while minimizing the probability of colliding with obstacles". In that case, in the open areas where there are no narrow passages, the system bypasses the belief node stabilizations. It speeds up and does not waste time gathering information and reducing its uncertainty as there is not much benefit in doing so in obstacle-free regions. However, once it faces with obstacles or narrow enough passages, it automatically decides to perform stabilization (partially) until the uncertainty is shrunk enough to safely traverse the narrow passage. Fig. 2.1, shows this phenomenon pictorially.

2.2.2 Results

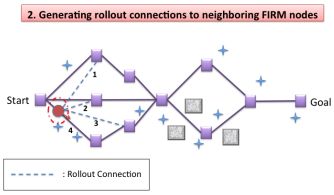
We proceed to present results on comparison of Rollout-based planning with the standard FIRM algorithm, followed by comparisons against an existing state-of-the-art optimization-based motion planning method.

2.2.2.1 Comparison with FIRM

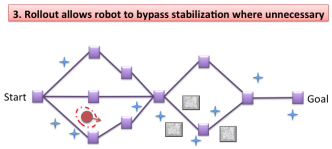
50 Monte Carlo simulations were carried out for a 2D navigation problem with the standard FIRM algorithm and with the proposed rollout-based method in belief space. The environment in Fig. 2.2(a) represents a 2D office space with obstacles



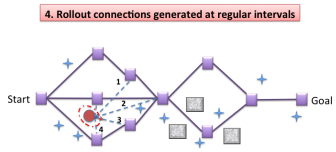
(a) A simple scenario with a FIRM roadmap, robot and environment as depicted.



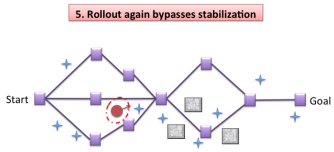
(b) The rollout policy is computed periodically. Four candidate edges are compared (three of them are shown in dashed line and the last one is the rest of the current edge.)



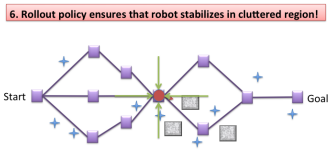
(c) In clutter-free regions, rollout takes a shorter route (edge 3), increasing performance and speed while loosing certainty (i.e., skipping node stabilization).



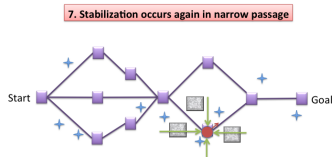
(d) While completing edge 3, the new rollout further cuts down task execution time by taking shorter route through a newly computed rollout edge 2.



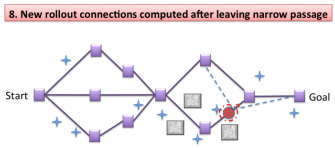
(e) The robot is approaching the cluttered region. As needed the planner will slow the robot down to trade performance with certainty.



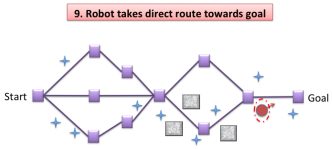
(f) Stabilization reduces localization uncertainty (covariance shrinks), thus guaranteeing high success probability.



(g) Stabilization occurs again as robot is still passing through the narrow passage.

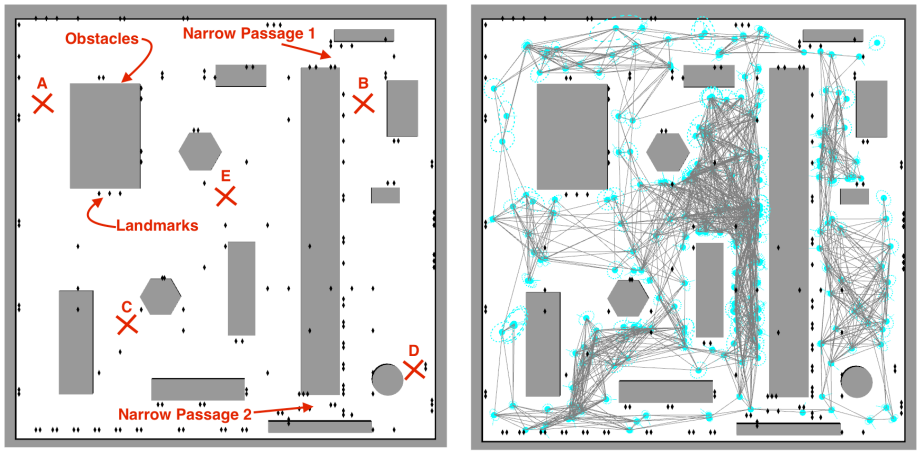


(h) New rollout connections allow bypassing stabilization.



(i) The robot approaching the goal.

Figure 2.1: A representational scenario depicting how rollout-based FIRM achieves higher performance compared to the standard FIRM algorithm while guaranteeing robustness. The 9 scenes depict different stages of task execution as the robot moves from the start to goal location.

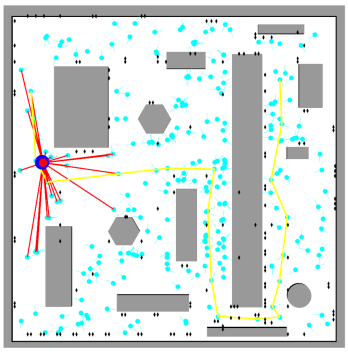


(a) (b)

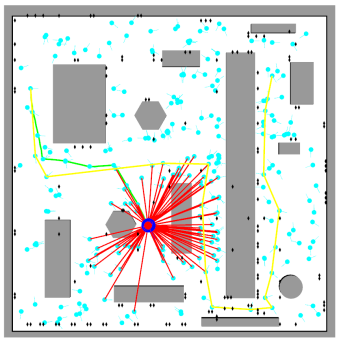
Figure 2.2: (a) The simulation environment. The black diamonds depict the landmarks, the grey polygons are the obstacles and the white space represents the free space. The locations of interest that the robot is tasked to visit are marked by red crosses. The two narrow passages P1 and P2 are marked, these represent regions of high collision probability (risky) due to the small clearance. (b) The underlying FIRM roadmap, the grey lines depict edges and the cyan disks are the nodes, the dashed ellipses represent the stationary covariance of the FIRM nodes.

and beacons placed in it. The robot is represented by a circular disk of radius 1m. There are two narrow passages *P1* and P2 which represent paths or edges of low transition probability/high collision probability. The narrow passages are 1.25m wide thus offering a very small clearance for the robot to pass through. The robot is placed at starting location 'A' and tasked to visit 4 different locations in a sequential manner, these are marked as B, C, D and E in the environment. Thus, we split the task into 4 sequential phases: 1) $A \rightarrow B$, 2) $B \rightarrow C$, 3) $C \rightarrow D$, and 4) $D \rightarrow E$.

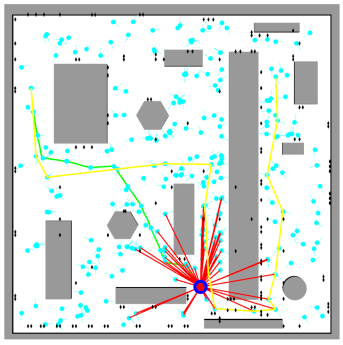
Rollout-based FIRM brings a level intelligence to this process of node stabilization. Rollout performs stabilization as and when required and bypasses it when possible. Thus, by bypassing the stabilization when not required, rollout allows the robot complete the task faster as well as with less execution cost. We now proceed



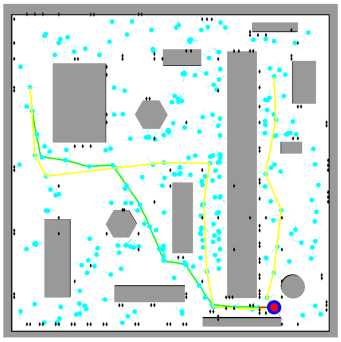
(a) The robot checks new connections with neighbors (red). The MLP under the nominal FIRM feedback is in yellow.



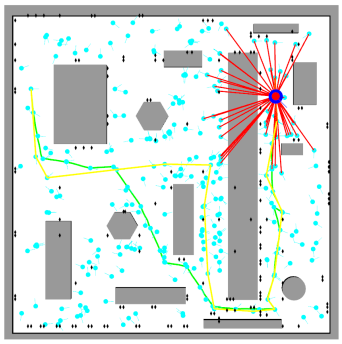
(b) Rollout guides the robot away from the MLP to a shorter and faster path (green). The new path is in a different homotopy class.



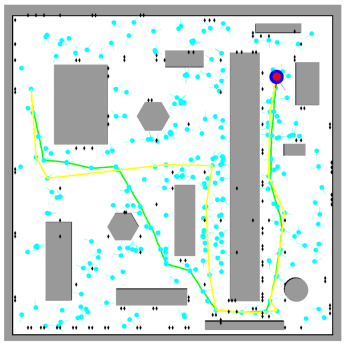
(c) Robot approaches narrow passage $P2$ through a more direct path as compared to the MLP.



(d) The robot stabilizes at a few FIRM nodes while passing through the narrow passage.

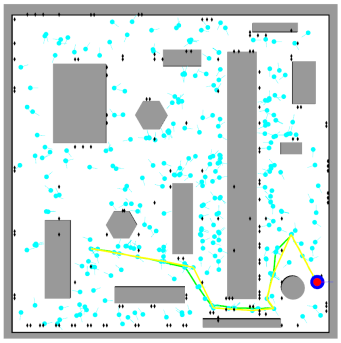


(e) The robot approaches goal location B.

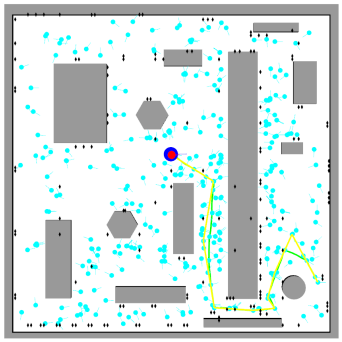


(f) The rollout path (green) is both shorter (geometrically) and faster (velocity-wise) than the MLP path under FIRM.

Figure 2.3: Segment 1 with rollout: Starting at A and going to B.



(a) Segment 3 ($C \to D$).



(b) Segment 4 ($D \to E$).

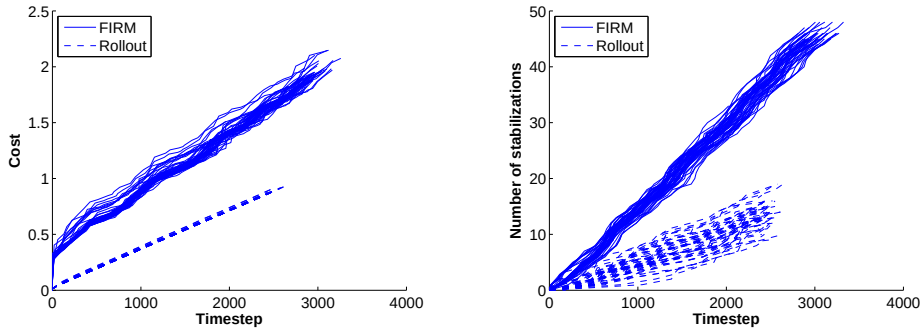
Figure 2.4: Asymmetric costs and random execution noises.

to discuss each aspect of the comparison in detail.

Cost of Execution: We recorded the amount of localization uncertainty (trace of covariance) along the robot's path. Figure 2.5(a) shows the cumulative version of this cost on 50 runs for the same task using rollout-based planner and with standard FIRM. We note that the cost for rollout based policy rises slower than the cost for FIRM, and as the planning horizon increases, rollout offers increasing returns in performance.

Selective stabilization: Node stabilization makes FIRM robust and scalable while maintaining the optimal sub-structure property on the graph (all the edges are independent of each other). Although stabilization allows FIRM to provide certain guarantees, it adds stabilization time and cost at each node to the time and cost of the mission. The rollout-based planner brings a higher level of intelligence to the process of node stabilization. Rollout performs stabilization when required and bypasses it when possible. Bypassing the stabilization allows the robot to complete the task faster and with less cost. Fig.2.5(b) shows the number of nodes the robot has stabilized to on 50 different runs. In this example, the robot stabilizes to ~ 45 nodes under FIRM compared to ~ 10 nodes under rollout-based planner ($\sim 75\%$ reduction), while the difference grows as the task becomes longer.

Time of Task completion: Another quantitative measure of performance is the time it takes for a planner to complete the task while guaranteeing a high likelihood of success. From Fig. 2.5(a) and 2.5(b), the time taken to complete the task with rollout is around 2500 time-steps (250 seconds) compared to 3000 time-steps (300 seconds) for FIRM. There is $\approx 15\%$ reduction in the time to complete the task compared to the standard FIRM algorithm. The improvement in execution time makes the rollout-based planner a better candidate than FIRM for time-sensitive applications.



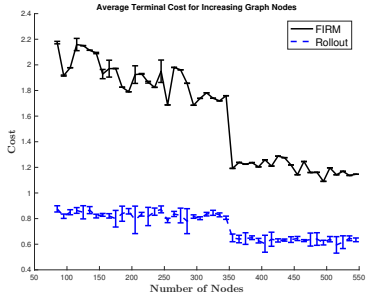
(a) (b)

Figure 2.5: Performance comparison of the original FIRM algorithm and rollout-based planner on 50 runs. (a) Cost of execution: The execution cost for FIRM rises faster than the cost of rollout based policy. (b) The number of belief nodes that the robot stabilizes to, during plan execution, which is consistently lower for the rollout-based planner.

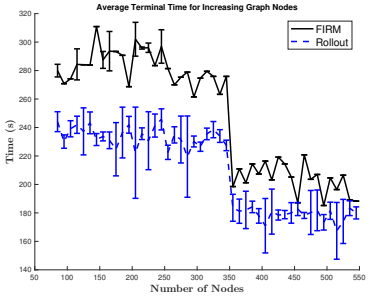
2.2.2.2 Comparison with State-of-the-Art

We compare our proposed method with iterative local optimization-based methods. As a representative method, we select the work of [145], which is a belief space variant of Iterative Linear Quadratic Gaussian Control [140] (referred to as BSP-iLQG, here). In BSP-iLQG, first, a tree-based approach (e.g., RRT) finds an initial solution to guide the robot from start to goal, and then BSP-iLQG solves a local optimization to compute the best local trajectory. In our comparisons we use a holonomic 2D robot and point beacon observation model (Sec. 6.2.2 [145]) whose signal strength decreases quadratically with distance.

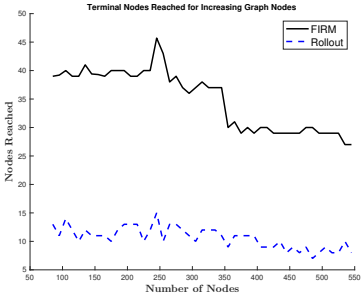
The optimization in BSP-iLQG is sensitive to the initial solution computed by RRT. In particular, the final solution is limited to the homotopy class of the initial solution. In environments like 2.7(a), this is not an issue because the globally optimal solution and the local optimal solution are the same. In this environment the solution of BSP-iLQG and the most-likely path (MLP) of the proposed method are



(a) The cost to complete the task reduces as the number of underlying graph nodes increases since more options are available. Sharp dips in the graph correspond to cases where adding a new node captures a new low-cost homotopy class.



(b) Time taken by the robot to complete the mission. As graph density increases robot finds more nodes to connect to during rollout and thus can take more shortcuts which reduces its total driving time.

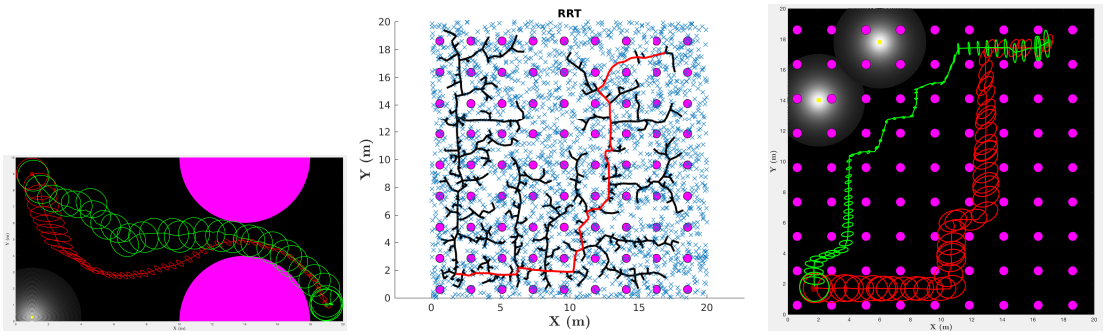


(c) The number of visited nodes (the number of stabilized) in Rollout is significantly smaller than FIRM.

Figure 2.6: The effect of increasing FIRM graph density on the Rollout solution behavior. The neighborhood connection strategy used to generate the graph connects each node to all nodes within radius R ($R = 5$ in these simulations). As the number of nodes in the graph crosses 350, a new connection is found through narrow passage 2 this leads sharp changes in the trends.

similar. However, in environments with more complex information distribution or with multiple homotopy classes (i.e., with multiple local minima), the sensitivity to the initial solution might degrade the performance. Fig. 2.7 shows one such example from 50 Monte Carlo runs. The initial RRT as shown in Fig. 2.7(b) computes a path that takes the robot towards the goal diagonally, whereas the optimal solution is quite different since the beacons are in the upper left corner of the environment.

The proposed rollout-based method does not require an initial solution, and will be able to find the optimal homotopy spanned by the underlying graph. This key difference is shown in Fig. 2.7(c). In addition, it can update the homotopy class during the execution based on potential deviations from the nominal homotopy class. In addition to the solution quality, the replanning time in optimization-based methods



(a) Simple environment with a single local/global minimum. The locally optimized solution (red) and path under Rollout (green).

(b) The RRT-based initial guess for local optimization-based methods.

(c) The locally optimized solution (red) and path under Rollout (green).

Figure 2.7: Comparison of Rollout vs local optimization-based methods. (a) A simple environment with one local/global minimum. In this environment, the solution of the local optimizer is the same as the global solution. (b) Local optimization-based methods require an initial solution. BSP-iLQG uses an RRT to find an initial solution. The red trajectory is an initial guess generated by RRT for the forest environment. The magenta circles depict the obstacles. (c) The final solution computed by BSP-iLQG (red) is restricted to a single homotopy class of solutions (local minima) whereas the Rollout-based plan (green) guides the robot closer to beacons, which is more informative by exploiting the global feedback under FIRM. The yellow spots are beacons whose signal strength declines quadratically as distance to a beacon increases. Thus farther the robot is from the yellow spots in the top-left corner, the less accurate its sensor measurements.

grows with the problem horizon. For example, in BSP-iLQG, the complexity of the optimization algorithm is in the order of $\mathcal{O}(RRT^*)+\mathcal{O}(N_l N_i)$, where $\mathcal{O}(RRT^*)$ refers to the complexity of the deterministic-based motion planning algorithm (e.g., RRT*) to create an initial solution and $\mathcal{O}(N_l N_i)$ refers to the complexity of optimization. N_l is the trajectory length and N_i is the number of iterations for the optimization to converge.

2.3 Discussion

A significant body of work exists on the problem of SLAP, however real-time online replanning has remained a challenging problem. This can be attributed to the fact that solutions of most methods depend on the initial belief, thus deviations from the local plan may necessitate re-computation of the entire solution which may not be feasible online. FIRM [10] was one of the first approaches to develop a belief space PRM that preserves the optimal substructure property on the graph. However FIRM suffered during online execution of the plan due to requirement of stabilization at each FIRM node in the FIRM policy. In Section 2.2 we described an extension to FIRM [10] that overcomes some of the shortcomings of FIRM for real-time replanning using a roll-out based approach. The proposed algorithm is able to switch between different homotopy classes of trajectories in real-time. It also bypasses the belief stabilization process of the state-of-the-art FIRM framework. By re-using the costs and transition probabilities computed in the offline construction phase of FIRM, the method is able to enable SLAP, via online replanning, in the presence of changes in the environment and large deviations in the robot's pose. Via extensive simulations we demonstrated performance gains when using the rollout-based belief planner. Such replanning is an important ability for practical systems where stochasticity in the system's dynamics and measurements can often result in failure. Comparisons with a state-of-the-art optimization based method showed that where optimization-based methods may get restricted to a local minima due to the initial guess, our approach is able to exploit the global feedback from FIRM to compute informative paths as it is independent of an initial guess. This is an important advantage in problems where solutions can belong to more than one homotopy class. The proposed SLAP solution takes an important step towards bringing belief space planners to physical applications and

advances long-term autonomy for mobile robots.

A majority of belief space methods have relied on a Gaussian representation for the robot's belief. Methods that have studied the problem of motion planning for non-Gaussian beliefs have assumed belief modes that are not widely separated in configuration space. While these methods may work well in most scenarios they are unable to deal with situations where ambiguity in data association leads to widely separated belief modes, e.g., symmetric environments. In the next chapter, we proceed to describe our novel approach to motion planning for disambiguating widely separated beliefs in the presence of data association ambiguity.

3. MOTION PLANNING FOR ACTIVE DATA ASSOCIATION AND LOCALIZATION IN NON-GAUSSIAN BELIEF SPACES

In practical mobile robot motion planning problems, situations may arise where data association between what is observed and the robot's map leads to a multimodal hypothesis on the state, for example a kidnapped robot with no a priori information or a mobile robot operating in a symmetric environment (see Fig. 3.1). Figure 3.1 depicts a problem wherein belief (the probability distribution over all possible robot states) modes are widely separated in an environment with symmetry. In such cases if a robot begins with an equal likelihood on all hypothesis, it is difficult to ascertain the true hypothesis as local sensing may result in identical information for all belief modes. Thus in practice a robot often has to seek information that helps to disambiguate its belief.[1]

Simply relying on randomized actions to correctly recover robot pose is known to be unreliable and inefficient in practice [50]. Further, existing methods to disambiguate multimodal hypothesis [50, 64, 53] rely on heuristics-based strategies (e.g., picking random targets, wall following etc.) to seek disambiguating information. As opposed to [50, 64, 53], our approach disambiguates, i.e., rejects incorrect hypothesis in a multimodal belief by actively seeking maximally disambiguating information in the map for each mode, and recovers the robot pose with a higher certainty threshold than current state-of-the-art.

Our Multi-Modal Motion Planner (M3P) achieves disambiguation in a multimodal belief by first finding a neighboring location (referred to as target state) for each belief mode and then creating a candidate action to guide the belief mode

[1]Parts of this material were previously published in [125].

to its target state such that these actions lead to information gathering behavior. The target states are chosen such that different modes of the robot's belief are expected to observe distinctive information at the target locations, thus accepting or rejecting hypotheses in the belief. We represent a multimodal hypothesis with a Gaussian Mixture Model (GMM) and use an Extended Kalman filter (EKF) based Multi-Hypothesis Tracking (MHT) approach to propagate the belief [122, 64, 124]. The main contributions of this chapter can be summarized as follows; (i) we develop a novel method for picking target states and creating candidate trajectories for a multimodal belief, our method then chooses a candidate trajectory such that maximum disambiguating information is observed which helps in rejecting incorrect hypotheses, (ii) we prove that under certain realistic assumptions, through a process of iterative hypothesis elimination, our method can localize to the true robot pose, (iii) we demonstrate an application in which a kidnapped ground robot is tasked to recover its pose.

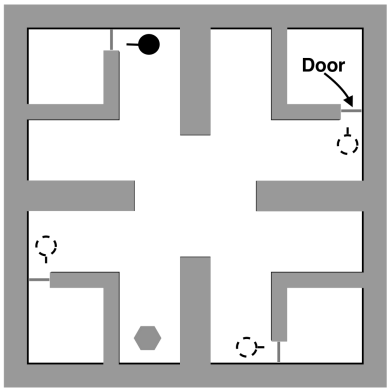


Figure 3.1: A scenario depicting a multi-hypothesis localization problem with widely separated modes in a world with 4 rooms with identical doors. The true hypothesis is depicted by the solid black disk, whereas others are depicted by dashed circles. As the robot cannot distinguish between the doors, all hypotheses are equally likely.

We motivate this chapter with the kidnapped robot scenario since it is one of the hardest localization problems and measures the ability of an algorithm to recover from global localization failures [137]. However, the method proposed is general

and can be extended to any planning situation where a multimodal belief arises in the robot state due to ambiguous data associations (a common practical issue in robot localization [137]). In Section 3.1 we state some preliminaries followed by the problem description. In Section 3.2 we present our method followed by experimental results in Section 3.3.

3.1 Preliminaries and Problem

Let C be the configuration space and $C_{free} \subset C$ be the set of collision free configurations. Let $x_k \in \mathbb{X}$, $u_k \in \mathbb{U}$, and $z_k \in \mathbb{Z}$ represent the system state, control input, and observation at time step k respectively. $\mathbb{X}$, $\mathbb{U}$, and $\mathbb{Z}$ denote the state, control, and observation spaces respectively. It should be noted that in our work, the state x_k refers to the state of the mobile robot, i.e., we do not model the environment and obstacles in it as part of the state. The non-linear state evolution model f and measurement model h are denoted as $x_{k+1} = f(x_k, u_k, w_k)$ and $z_k = h(x_k, v_k)$, where $w_k \sim \mathcal{N}(0, Q_k)$ and $v_k \sim \mathcal{N}(0, R_k)$ are zero-mean Gaussian process and measurement noise, respectively. The belief b_k at time t_k can be represented by a Gaussian Mixture Model (GMM) as a weighted linear summation over Gaussian densities. Let $w_{i,k}$, $\mu_{i,k}$ and $\Sigma_{i,k}$ be the weight, mean vector, and covariance matrix associated to the i^{th} Gaussian $m_{i,k}$ respectively at time t_k, then $b_k = \sum_{i=1}^{M_k} w_{i,k} m_{i,k}, \;\; m_{i,k} \sim \mathcal{N}(\mu_{i,k}, \Sigma_{i,k})$, where M_k is the number of modes at time t_k. We state our problem as follows:

Given an a priori map, system dynamics and observation models, construct a belief space planner $G(b_k)$ such that under the planner G, given an initial multimodal belief b_0, the sequence of future observations allow a robot to localize about its true pose.

Note that there may be degenerate cases, where the map may not allow actions that lead to hypothesis elimination such that the belief converges to a unimodal

distribution (e.g., in a map with two identical closed rooms, if a robot is kidnapped and placed in either room, it cannot distinguish which room it is in). In such cases, M3P attempts to minimize the number of modes M_k (by design), but it is not possible to pre-compute what this minimum value of M_k is without explicit knowledge of the true hypothesis [41] in a multimodal belief.

3.2 Methodology

We begin by defining certain key concepts used in the M3P planner.

Uniqueness Graph: A graph U_g, whose nodes are states sampled from the collision free space C_{free} and whose edges relate the similarity of information observed at the sampled locations.

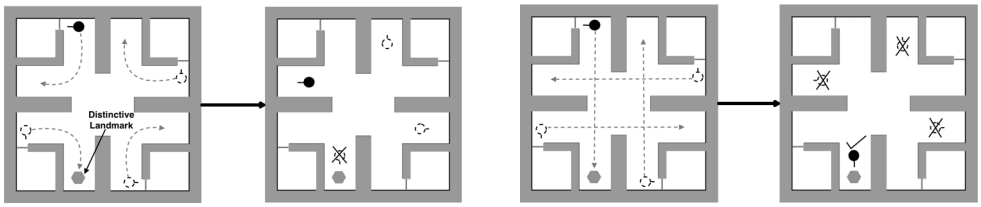
Target State: A target state $v_i^{tt} \in U_g$ for mode m_i is a node of the uniqueness graph which belongs to some neighborhood of radius R of the mode's mean μ_i such that if each mode were to visit its target, the observations at the target would lead to disambiguation in the belief.

Candidate Policy: A candidate policy π_i for mode m_i is a local feedback controller that guides the mode to its target v_i^{tt}.

The M3P methodology has two phases, an offline phase in which we generate U_g and an online phase in which we use the offline computations and plan in a receding horizon manner to disambiguate the belief.

3.2.1 Computing the Uniqueness Graph: Offline Phase

The uniqueness graph U_g is constructed by uniformly sampling the configuration space and adding these samples as nodes of U_g. Once a node is added, we simulate the observation for the state represented by that node. Let v_α be one such node and z^{v_α} be the observation if the robot were to be in state v_α. We add an edge $E_{\alpha\beta}$ (undirected) between two nodes v_α and v_β if the simulated observations at both



(a) Candidate A leads to negative information for the mode in lower left corner. It expects to see the distinctive landmark which robot doesn't observe, and is thus rejected.

(b) Candidate B leads the true hypothesis to be confirmed as the robot sees the distinctive landmark.

Figure 3.2: Extending the example in Fig. 3.1, we depict 2 candidate trajectories; candidates A & B and the effect of their execution. Candidate A leads to one disambiguation and candidate B results in complete disambiguation. Candidate B is a better choice, however the difficulty of picking B lies in the fact that robot does not know its true hypothesis a priori.

nodes are similar. Further, the edges are weighted and the weight is dependent on the similarity in information observed, i.e., for edge $E_{\alpha\beta}$ the weight $\omega_{\alpha\beta} = \tau(z^{v_\alpha}, z^{v_\beta})$ where $\tau : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$ computes a measure of similarity between two observations. Note that the form of τ is general and can be changed to suit the problem domain (perception model). Figure 3.3 explains this concept visually for a landmark based observation model, where each landmark has some discrete signature (identifier) that a robot can detect. In Fig. 3.3 state v_α observes z^{v_α} with signatures ${}^s z^{v_\alpha} = \{s_1, s_2, s_3\}$, i.e., the landmarks with signature s_1, s_2 and s_3 and at v_β observes ${}^s z^{v_\beta} = \{s_1, s_2, s_4\}$, the edge weight $\omega_{\alpha\beta}$ for edge $E_{\alpha\beta}$ is $\omega_{\alpha\beta} = \tau(z^{v_\alpha}, z^{v_\beta}) = |{}^s z^{v_\alpha} \cap {}^s z^{v_\beta}| = |\{s_1, s_2\}| = 2$. A higher edge weight signifies that the states represented by the vertices of that edge are more likely to observe similar information. The lack of an edge between two nodes means that if a robot were to make an observation at those two states, it would see distinctly different information.

The complexity for the construction of U_g is $\mathcal{O}(n^2)$ (where n is the number of

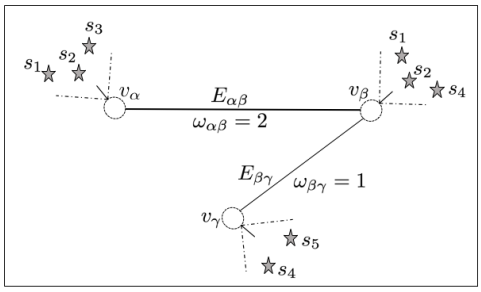


Figure 3.3: Simple example of a uniquness graph with 3 nodes $\{v_\alpha, v_\beta, v_\gamma\}$ and 2 edges $\{E_{\alpha\beta}, E_{\beta\gamma}\}$. The nodes v_α and v_γ do not see any similar landmark hence there is no edge between them. Here $\tau(z^{v_i}, z^{v_j}) = |{}^s z^{v_i} \cap {}^s z^{v_j}|$ for $i, j \in \{\alpha, \beta, \gamma\}$.

samples) as each sample (node) is checked with every other for information overlap. Due to its random nature, sampling may often occur in regions of low information density (e.g., regions where there are few or no landmarks). One can often circumvent this issue by increasing the number of samples. As U_g is computed offline, the online performance is not significantly affected. Recent work in [112] suggests a localization aware sampling strategy which may be explored in future work.

3.2.2 RHC based Planning: Online Phase

In a multimodal scenario, we claim that the best action to take is one that guides a robot without collision through a path that results in information gain such that a disambiguation occurs (one or more hypotheses are rejected, see Fig. 3.2). Algorithm 1 describes the online planning process. In step 3, the planner picks target states for each belief mode such that visiting a target can either prove or disprove the hypothesis. In step 4, the planner generates a set of candidate policies to drive each mode to its target. In step 5, the expected information gain for each policy is computed and we pick the best one, and in step 7, the multimodal belief is propagated according to the action and observations. We proceed to describe steps 3, 4, 5 and 7 of Algorithm 1 below.

Algorithm 1: M3P: MultiModal Motion Planner

```
1  Input: b
2  while b ≠ N(μ, Σ) do
3      {v^tt} ← Pick target states for belief modes (see Alg. 2);
4      Π ← Generate candidate policies to connect each mode to its target;
5      π* ← Pick optimal policy from Π;
6      forall u ∈ π* do
7          b ← Apply action u and update belief (see Alg. 3 for weight update
             calculation);
8          if Change in number of modes or Expect a belief mode to violate
             constraints then
9              break;
10 return b;
```

3.2.2.1 Picking the target state for a mode

Algorithm 2 describes in detail the steps involved to pick a target state for a belief mode. Let us pick a mode $m_{i,k} \sim \mathcal{N}(\mu_{i,k}, \Sigma_{i,k})$ from the belief. To find the target $v^{tt}_{i,k}$ for $m_{i,k}$, we first choose the set of nodes $N_{i,k} \in U_g$ (Section 3.2.1) which belong to the neighborhood of the mean $\mu_{i,k}$ at time t_k (steps 3 and 4, Alg. 2). Then, we find the target node $v^{tt}_{i,k} \in N_{i,k}$ which observes information that is least similar in appearance to that observed by nodes in the neighborhoods $N_{j,k}$ of all other modes $m_{j,k}$ where $j \neq i$. To do this, after computing $N_{i,k}$, we calculate the total weight of the outgoing edges from every node $v_{i,k} \in N_{i,k}$ to nodes in all other neighborhoods $N_{j,k}$ where $j \neq i$ (steps 7-13, Alg. 2). The node which has the smallest outgoing edge weight (steps 14-16, Alg. 2), is the target candidate $v^{tt}_{i,k}$ for $m_{i,k}$ as the observation $z^{v^{tt}_{i,k}}$ would be least similar to the information observed in the neighborhood of all other modes m_j where $j \neq i$.

Algorithm 2: Finding the target for i-th mode

```
Input: b_k, i , U_g
Output: v^tt_{i,k}
forall l ∈ [1, M_k] do
    N_{l,k} ← Find neighborhood nodes for μ_{l,k} in U_g;
minWeight ← Arbitrarily large value;
v^tt_{i,k} ← −1;
forall v ∈ N_{i,k} do
    w ← 0;
    for N_{j,k} ∈ {N_{1,k}, ..., N_{M_k,k}} \ N_{i,k} do
        forall e ∈ Edges connected to v do
            forall p ∈ N_{j,k} do
                if p is a target of edge e then
                    w ← w + edgeWeight(e);
    if w < minWeight then
        minWeight ← w;
        v^tt_{i,k} ← v;
return v^tt_{i,k};
```

3.2.2.2 Generating candidate policies for belief modes

Once the targets corresponding to each mode have been picked, we need to find the control action that can take a mode from its current state to the target state. We generate the candidate trajectory that takes each mode to its target using the RRT* planner [73]. Once an open loop trajectory is computed, we generate a local policy π_i (feedback controller) for the i-th mode, which drives the i-th mode along this trajectory. Let Π be the set of all such policies for the different modes.

3.2.2.3 Picking the Optimal Policy

After generating the set Π of candidate policies, we evaluate the expected information gain ΔI_i for each policy π_i and pick the optimal policy π^* that maximizes

this information gain. We model this information gain as the discrete change in the number of modes. To compute the expected change in the number of belief modes, we simulate the most-likely belief trajectory, i.e., approximating noisy observations and actions with their most-likely values [15, 29, 117, 58]. The steps to calculate the expected information gain for a candidate policy $\pi_i \in \Pi$ are as follows:

1. For every belief mode $m_{j,k} \in b_k$.
   (a) Assume that robot is at $m_{j,k}$.
   (b) Simulate π_i and propagate all the modes.
   (c) Compute information gain $\Delta I_{i,m_{j,k}}$ for π_i.
2. Compute the weighted information gain $\Delta I_i = \sum_{j=1}^{M_k} w_{j,k} \Delta I_{i,m_{j,k}}$.

After computing the expected information gain for each policy, we pick the gain maximizing policy. The computational complexity of this step is $\mathcal{O}(M_k^3 L_{max})$ (where M_k is the number of belief modes and L_{max} is the maximum candidate trajectory length). This is due to the fact that each policy is simulated for each mode for the length of policy, where at every step of policy execution, there are M_k filter updates.

3.2.2.4 Belief Propagation Using GMM

We first discuss our decision to use EKF based MHT over a particle filtering approach. In practical localization problems, a relatively small number of Gaussian hypotheses are sufficient for maintaining the posterior over the robot state, secondly the filtering complexity grows linearly in the number of hypotheses and finally due to the computational complexity of picking the optimal policy (see previous section), the number of samples required for a particle filter would make re-planning significantly harder.

Now, we proceed to describe the belief update step which propagates each mode's mean and covariance and determines how likely each mode is in the belief. The mean and covariance of each belief mode is updated with its own Kalman filter, i.e., a parallel bank of Kalman filters is used. Each filter undergoes an identical prediction update and the measurement update is a function of the data association between robot's observation and most-likely observation for belief mode. Note that during belief propagation our estimate remains conservative in that covariance is not truncated when passing through obstacles. In this regard [111] provides a direction for future enhancements; the method of [111] truncates estimation covariance by first applying an affine transformation to the 2D Gaussian and then truncating the 1D distribution according to given linear inequality constraints. The weights $w_{i,k}$'s are updated based on the measurement likelihood function as

$$w_{i,k+1} = w_{i,k} e^{-\frac{1}{2} D^2_{i,k+1}}, \tag{3.1}$$

where $D_{i,k+1}$ is the Mahalanobis distance between the sensor observation and most-likely observation for mode m_i such that

$$D^2_{i,k+1} = (z_{k+1} - h(\mu_{i,k+1}, 0))^T R_k^{-1} (z_{k+1} - h(\mu_{i,k+1}, 0)). \tag{3.2}$$

The weights are normalized such that $\sum_{i=1}^{M_k} w_{i,k+1} = 1$.

A known issue with EKF-based MHT is that a naive implementation is unable to process negative information [137]. Negative information refers to the lack of information which one may expect to see and can certainly help in disproving a hypothesis (see Fig. 3.2(a)). We now proceed to describe how negative information is factored into the weight update.

Factoring Negative Information: Depending on the state of the robot, individual hypotheses and data association results, we might have several cases. We discuss this issue in the context of a landmark based measurement model. At time t_{k+1}, let $n_{z_{k+1}}$ be the number of landmarks observed by the robot and $n_{z^p_{i,k+1}}$ be the number of landmarks that we predict to see for m_i where $z^p_{i,k+1} = h(\mu_{i,k+1}, 0)$ is the predicted observation. Then $n_{z_{k+1}} = n_{z^p_{i,k+1}}$ means that the i-th mode expected to see as many landmarks as the robot observed; $n_{z_{k+1}} > n_{z^p_{i,k+1}}$ implies the robot observes more landmarks than predicted for the mode; $n_{z_{k+1}} < n_{z^p_{i,k+1}}$ implies the robot observes less landmarks than predicted for the mode. Also, we can have the number of data associations to be less than the number of predicted or measured observations or both. This means that we may not be able to make a unique association between each predicted and observed landmark. At time t_{k+1}, we estimate the Mahalanobis distance $D_{i,k+1}$ (Eq. 3.2) for mode m_i between the predicted and observed landmarks that are matched by the data association module and update weight according to Eq. 3.1. Then we multiply the updated weight by a factor γ, which models the effect of duration $\beta_{i,k+1}$ for which the robot observes different landmarks than the i-th mode's prediction; and the discrepancy α in the number of data associations. When a belief mode is initialized, we set $\beta_{i,0} = 0$. The weight update procedure is described in Algorithm 3. After each weight update step, we remove modes with negligible contribution to the belief, i.e., when $w_{i,k+1} \leq \delta_w$ where δ_w is user defined.

3.2.3 Analysis

In this section, we show that under certain assumptions on the structure of the environment, the receding horizon planner M3P can guarantee that an initial multi-modal belief is driven into a unimodal belief in finite time. We now proceed to state our assumptions.

Algorithm 3: GMM Weight Update

Input: $w_{i,k}, \mu_{i,k+1}, \beta_{i,k}, \delta t$
Output: $w_{i,k+1}, \beta_{i,k+1}$
$z_{k+1}, n_{z_{k+1}} \leftarrow$ Get sensor observations;
$z^p_{i,k+1}, n_{z^p_{i,k+1}} \leftarrow$ Get predicted observations for $\mu_{i,k+1}$;
$n_{z_{k+1} \cap z^p_{i,k+1}} \leftarrow$ Do data association;
$w'_{i,k+1} \leftarrow$ Update and normalize weight according to likelihood function;
$\gamma \leftarrow 1$;
if $n_{z^p_{i,k+1}} \neq n_{z_{k+1}}$ *or* $n_{z^p_{i,k+1}} \neq n_{z_{k+1} \cap z^p_{i,k+1}}$ **then**
    $\alpha \leftarrow max(1 + n_{z_{k+1}} - n_{z_{k+1} \cap z^p_{i,k+1}}, 1 + n_{z^p_{i,k+1}} - n_{z_{k+1} \cap z^p_{i,k+1}})$;
    $\beta_{i,k+1} \leftarrow \beta_{i,k} + \delta t$;
    $\gamma \leftarrow e^{-\alpha \beta_{i,k+1} 10^{-4}}$;
else
    $\beta_{i,k+1} \leftarrow 0$;
$w_{i,k+1} \leftarrow w'_{i,k+1} \gamma$;
return $w_{i,k+1}, \beta_{i,k+1}$;

Assumption 1 *For every mode m_i, the environment allows for the existence of some target state v_i^{tt} and some homotopy class of paths through which the robot can visit v_i^{tt} if the robot is actually at mode m_i.*

Assumption 2 *If the robot is actually at mode m_i, and its associated target state is v_i^{tt}, let $B_r(v_i^{tt})$ to be a neighborhood of radius $r > 0$ centered at the target v_i^{tt} such that if robot state $x \in B_r(v_i^{tt})$ (robot in vicinity of target), exteroceptive observations can confirm that m_i is the true hypothesis.*

Due to the uncertain nature of the actuation and sensing process, the existence of a path to visit a target location does not guarantee that a robot can drive its belief along this path or that on reaching neighborhood $B_r(v_i^{tt})$, localization uncertainty will be sufficiently low so as to make a disambiguating data association. Let the true belief be mode m_i. Let $F \subset C \setminus C_{free}$ be the set of failure states, and let L be

the finite stopping time for policy π_i defined as the time at which collision occurs or the belief mean μ_i reaches the neighborhood $B_r(v_i^{tt})$. Denote $P^{\pi_i}(x_L \in F|m_i)$ as the probability that policy π_i drives the underlying state x into a collision given the initial belief is m_i.

Assumption 3 *Given Assumption 1, let mode $m_i \sim \mathcal{N}(\mu_i, \Sigma_i)$, with $||\Sigma_i|| < \bar{P} < \infty$ (initial covariance is bounded) be the true hypothesis. We assume that under the feedback policy π_i, the failure probability $P^{\pi_i}(x_L \in F|m_i)$ is sufficiently low such that we can drive the robot state x into the neighborhood $B_r(v_i^{tt})$ with a high probability $\int_{B_r(v_i^{tt})} p^{\pi_i}(x_L|m_i, \neg F)dx > 1 - \delta$ for any $\delta > 0$ where $p^{\pi_i}(x_L|m_i, \neg F)$ is the terminal pdf on the state under policy π_i when the robot does not collide.*

Assumption 4 *The environment (world) in which the robot operates is static.*

Proposition 1 *Under Assumptions 1, 2, 3 and 4, given any initial multimodal belief $b_0 = \sum_i w_{i,0} m_{i,0}$, the receding horizon planner M3P drives the belief process into a unimodal belief $b_T = m_T \approx \mathcal{N}(\mu_T, \Sigma_T)$ in some finite time T.*

Proof 1 *Given an initial belief b_0, let π_{i^*}, i.e., candidate policy for mode m_{i^*}, be the one that results in most information gain as required by M3P. We have only two possibilities; (i) Case 1: Mode m_{i^*} is the true hypothesis, or (ii) Case 2: Mode m_{i^*} is not the true hypothesis. If case 1 is true, due to Assumptions 1, 2 and 3, M3P can confirm that m_{i^*} is the true hypothesis by visiting the target location and rejecting all other hypotheses in the process (see Fig. 3.2(b)). If case 2 is true then the robot is at some other mode m_j where $j \neq i^*$. In case 2, as policy π_{i^*} is executed, two situations can arise, either (i) π_{i^*} is executed fully in which case m_{i^*} will expect to see distinctive information at its target location which the robot will not observe, leading to a disambiguation immediately due to negative information (see*

Fig. 3.2(a)) or (ii) the policy π_{i^*} becomes unfeasible at some point of its execution in which case we immediately know that the robot is not at mode i^* since we know that the map did not change during the execution of π_{i^*} (Assumption 4) and thus, there is a disambiguation whereby mode i^* is discarded. Thus we see that either π_{i^*} confirms the true hypothesis or the number of modes is reduced by at least one. After this disambiguation, we restart the process as before and we are assured that at least one of the modes is going to be disambiguated and so on. Thus, it follows given that we had a finite number of modes to start with, the belief eventually converges to a unimodal belief. Further, since each of the disambiguation epochs takes finite time, a finite number of such epochs also takes a finite time, thereby proving the result.

Remarks: The above result shows that the M3P algorithm will stabilize the belief process to a unimodal belief under Assumptions 1, 2, 3 and 4. In the case that Assumption 1 is violated we are either (i) unable to find a target which allows the robot to observe distinctive information (e.g., trivial case of a robot operating in a world with identical infinite corridors) or (ii) we may find such a target but the environment geometry does not allow for any path to visit it (e.g., robot stuck in one of many identical rooms and the doors are closed). These violations refer to degenerate cases that rarely occur in practical motion planning problems. Assumptions 2 and 3 can be violated when all candidate trajectories pass through regions lacking enough information, either because the region is unknown or featureless. In such a case the localization uncertainty on each mode may grow so high that we cannot make data associations at the target location to disambiguate the multimodal belief. Thus these two assumptions imply that the known map has enough information sources (see Fig. 3.4). Handling the issue of maps that are either unknown, partially known or sparse in information sources is beyond the scope of this paper and presents an

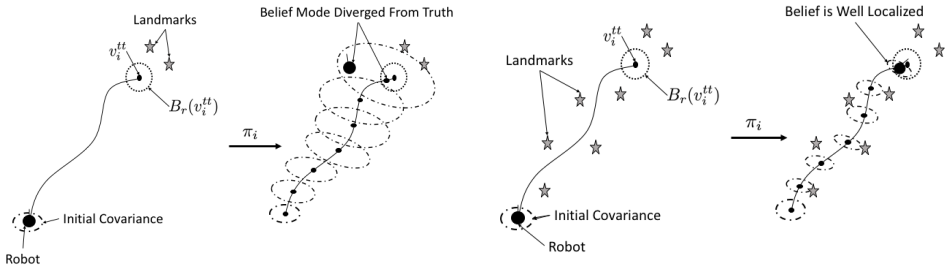


Figure 3.4: Evolution of the true belief mode in environments with and without sufficient information. (a) No landmarks present along the candidate trajectory, leading to high uncertainty at the end. The belief mode has diverged from the robot pose and it is no more possible to make an accurate data association for the landmarks at the target. (b) Sufficient information along the candidate trajectory leads the belief mode to be well localized at the end, allowing unambiguous data association for the landmarks at the target.

important direction for future research. Assumption 4 (static world) is common in localization literature, though it may be violated in certain scenarios. In such cases, if the map is not changing rapidly, one may use sensory observations to incorporate new constraints into the map and trigger replanning.

3.3 Experimental Results

We present experimental results for two motion planning scenarios wherein the robot is placed randomly at a location in an environment which is identical to other locations in appearance[2]. Thus the initial belief is multimodal, the goal of the experiment is to use the non-Gaussian planner M3P described in Section 3.2 to localize the robot pose. We first describe the system setup to motivate the experiment followed by the results.

[2]In the interest of space we only present one experiment here; a supplementary video [6] is provided that clearly depicts every stage of both our experiments.

3.3.1 System Description

We used a low-cost Arduino based differential drive robot equipped with an Odroid U3 computer running ROS on Ubuntu 14.04 and an off-the-shelf Logitech C-310 webcam for sensing. The onboard computer uses a wifi link to communicate with the ground control station (laptop running ROS on Ubuntu 14.04). The ground station runs the planner and image processing algorithms while communicating with the robot via wifi. The kinematics of the robot are represented by a standard unicycle motion model. The observation model is a vision-based range bearing sensor augmented with appearance information (see [137] Sec. 6.6.2) such that $z_k = h(x_k, v_k) = [(r_1, \phi_1, s_1)^T, (r_2, \phi_2, s_2)^T, \dots]$ where r_l, ϕ_l, s_l are the range, bearing and signature for the l-th observed landmark. The signature is an integer value and identical landmarks have the same signature. For this observation model, the function τ (compute information overlap between two observations, see Sec. 3.2.1) is identical to that described in Fig. 3.3. In the real world, landmark appearances may change due to environmental conditions (e.g., lighting), perspective etc., which may adversely affect detection, such issues require more complex perception models and map representations which are outside the scope of this work.

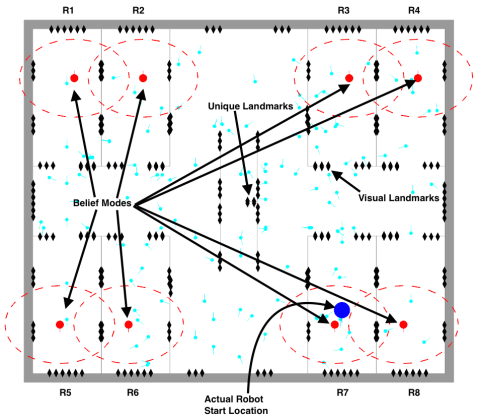
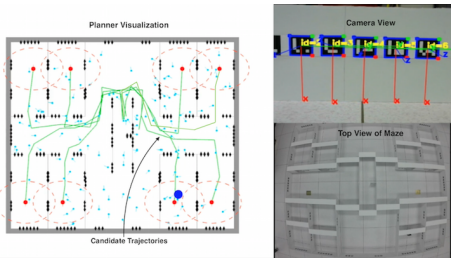
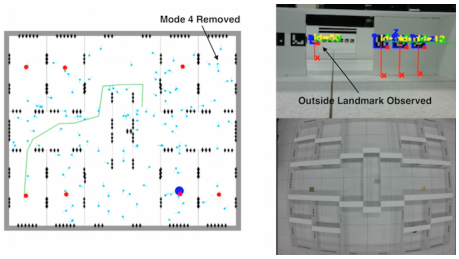


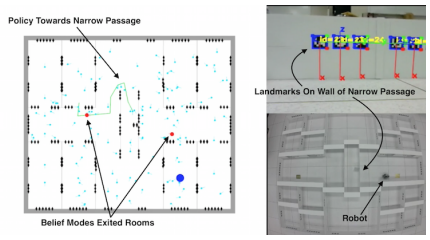
Figure 3.5: Environment with 8 rooms marked R1-R8 and belief at the start of first run. Robot is placed in room R7 (blue disk), initial sampling leads to 8 belief modes, one in each room. The black diamonds mark the locations of augmented reality markers in the environment. Unique landmarks are placed inside the narrow passage, such that if robot enters the passage from either side, it sees distinctive information.



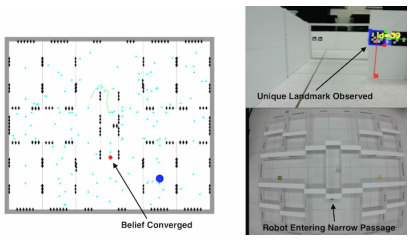
(a) The planner visualization showing the candidate trajectories (green). The top right image shows the view from the onboard camera, with the detected marker information overlaid. The bottom-right image shows the top-view of the maze in which the robot is run.



(b) Robot observes landmark ID 55 on the door of the opposite room causing the weights of modes $m_1, m_3, m_4, m_5, m_6, m_8$ to gradually decrease which leads to these modes being removed from the belief.



(c) Robot has exited the room and looks at the outside wall of the narrow passage. Modes m_2 and m_7 are symmetrically located due to information in the map observed by the robot.



(d) Belief mode has converged to the the true belief as the robot enters the narrow passage and observes the unique landmark (ID 39).

Figure 3.6: Snapshots of first run of the experiment at different times.

3.3.2 Scenario

We constructed a symmetrical maze that has 8 identical rooms (R1-8) as shown in Fig. 3.5. Augmented reality (AR) markers were placed on the walls which act as the landmarks detected by the vision-based sensing system of the robot [52]. When the robot sees a landmark, it can detect the range, bearing as well as its signature. To create ambiguity in the data association, we placed multiple AR markers with the same signature in different parts of the environment. For example, one of the

symmetries in our experiment is the inside of each room. Each room in the maze appears identical to the robot as markers with the same appearance are placed on each room's walls with an identical layout. Thus, if the robot is placed in a location with markers similar to another part of the environment, the data associations lead the robot to believe it could be in one of these many locations, which leads to a multimodal belief on the state. We also place four unique markers in a narrow passage in the center of maze as marked in Fig. 3.5. To successfully localize, the robot must visit this location in order to converge to its true belief.

The robot is initially placed in room R7 and not given any prior information about its state. To estimate initial belief b_0, we discretize the 2D environment with a grid cell size of 0.04m and at each grid point place four hypothesis separated in orientation by 90° starting at 0°. The initial covariance for each hypothesis is $\Sigma_0 = diag([0.04, 0.04, 0.04])$. For our maze with dimensions $4.56\text{m} \times 3.81\text{m}$, it results in 42408 initial hypothesis wherein only collision-free states are preserved and the rest are discarded. While the initial number of hypothesis appears to be large, it is important to have sufficient number of hypothesis in order to guarantee that at least one hypothesis' pdf sufficiently captures true robot state. After this, the robot remains stationary and the sensory measurements are used to update the belief state and remove the unlikely modes with weight $w \leq \delta_w = 0.01$. This process of elimination continues until we converge to a fixed number of modes. Figure 3.5 shows the initial belief. The robot plans its first set of candidate actions as shown in Fig. 3.6(a). After the candidates are evaluated, the policy based on mode m_5 in room R5 is chosen and executed. As the robot turns, it sees a landmark on the wall outside the room (shown in Fig. 3.6(b)). This causes mode m_4 to be deleted. Immediately, replanning is triggered and a new set of candidate trajectories is created. In successive steps, we see that first modes m_3 and m_5 are deleted and then after the next two

replanning steps, modes m_8, m_1 and m_6 are deleted. We notice that the robot does not move till only the 2 most-likely modes are remaining. The reason for this is that seeing the marker on the outside wall has the effect of successively lowering the weights of the unlikely modes. As the mode weights fall below the threshold, they are deleted, which triggers the replanning condition. Once the belief has converged to the two most-likely modes m_2, m_7 (as expected by the symmetry) a new set of candidate policies is created and the policy based on mode m_2 is chosen. This policy leads the modes out of the rooms, and towards the narrow passage. Figure 3.6(c) shows both belief modes executing the policy based on mode m_2. While executing this policy, replanning is triggered as the robot exceeds maximum horizon (60 secs) for policy execution. The final policy drives the robot into the narrow passage and the unique landmarks are observed (Fig. 3.6(d)) which leads the belief to converge to the robot pose.

3.3.3 Notes

Our approach results in a behavior which guides the robot to seek disambiguating information. Candidate trajectories are regenerated every time a belief mode is rejected or a constraint violation is foreseen and time to re-plan reduces drastically as number of modes reduce. Thus, first few actions are the hardest which is to be expected as we start off with a large number of hypotheses. Finally, the planner is able to localize the robot safely.

In the method of [41], once an observation is made, the simulated robot retraces its path to reset odometery-based estimation error; this may be inefficient in practical scenarios due to limited on-board power. Further, as opposed to [41] our robot is not equipped with a global heading sensor. In case a global orientation sensor is present, fewer hypothesis may be required due to the additional constraint of known

heading. In [50], the authors showed in simulation that random motion is inefficient and generally incapable of localizing a robot within reasonable time horizons especially in cases with symmetry (e.g., office environments with long corridors and similar rooms). In [64] the authors consider a physical robot localized in their experiments when one mode attains a weight ≥ 0.8, in contrast our approach is more conservative in that we only consider the robot localized when a mode has weight ≥ 0.99. We can afford to be more conservative as our localization strategy actively seeks disambiguating information using prior map knowledge as opposed to a heuristic based strategy. While our experiment acts as a proof of concept, there are certain phenomenon such as cases where the belief modes split into child modes, or dynamic environments which were not covered and will be addressed in future work.

3.4 Discussion

We studied the problem of mobile robot motion planning for active data association in order to correctly localize a robot when the initial underlying belief is multimodal (non-Gaussian). Our main contribution is a planner M3P that generates a sequentially disambiguating policy through active data association, which leads the belief to converge to the true hypothesis. We are able to show in practice that the robot is able to recover from a kidnapped state and localize in environments that present ambiguous data associations such that the underlying belief modes are widely separated. Compared to previous works, we take a non-heuristic approach to candidate policy generation and selection, while remaining conservative in accepting the true hypothesis. A current limitation may be the computational cost for the policy selection step in large maps which lead to a high number of hypotheses.

Chapters 2 and 3 investigated the problem of motion planning under uncertainty when a map is given. However, to achieve true autonomy a robot must be able to

build a map of its environment using its on-board sensors. In the proceeding Chapter we discuss relevant related work in the field of SLAM followed by our contributions to the state-of-the-art.

4. A REVIEW OF SIMULTANEOUS LOCALIZATION AND MAPPING

SLAM has a rich body of prior work and it would be hard to do justice to all the contributions made by researchers before us. Our objective is to provide a broad overview of key contributions in the field of SLAM with a particular focus on works that directly relate to the contributions in this dissertation. In the next two sections we first survey filtering-based SLAM methods and look into methods that leverage relative feature measurements to solve the SLAM problem. Next we review smoothing-based SLAM methods followed by SPLAM (also sometimes referred to as active SLAM in literature). Finally in Section 4.4 we look at the role of orientation (heading) in SLAM as it is the key to consistently estimating map and robot belief.

4.1 Filtering-based SLAM Methods

Initial work in [133] showed that an EKF-based approach can be used to estimate robot and map state jointly online. In [38] the authors analyze the convergence properties of EKF-SLAM. In [54], using a simple 1-D analysis the authors show the importance of observing "good" features as opposed to mapping a large number of features. The works of [67, 28, 60, 13] analyzed the consistency of EKF-SLAM to determine the potential drawbacks and limitations. In [67], the authors show that even for a simple case of a stationary robot with no process noise, heading estimate drifts and tends to become overconfident for a 2D robot. The general consensus is that in the long-term, EKF-SLAM tends to become overconfident and the uncertainty estimate does not reflect the true error. Further, it is the heading estimate uncertainty which is a major cause of inconsistency and catastrophic failure. The analysis of [13] showed that if periodic heading fixes are provided then small heading uncertainty leads to consistent results for longer durations and stabilizing noise

can be added to make the estimates less overconfident. In [59] the authors analyze observability properties of EKF-SLAM and find conditions for observability to be satisfied (i.e., only the position is observable). These conditions are then used to solve for the optimal linearization points such that the observability and minimum posterior error condition are both balanced, this is called Observability Constrained EKF (OC)-EKF. Another formulation First Estimates Jacobian (FEJ)-EKF is also discussed. In FEJ-EKF, Jacobians are calculated for the feature locations when they are first observed. Both of these are shown to be consistent and perform much better than standard EKF-based SLAM. FEJ-EKF has a potential drawback that if the first estimate was corrupt, then linearization is spurious. To overcome the computational complexity of the standard EKF-SLAM algorithm an information filter-based approach with constant time updates was developed and analyzed in [138, 46], however this approach too faces consistency issues due to the sparsification employed. The work in [105] presents a method called Joint Compatibility Branch and Bound (JCBB) for data association in stochastic mapping in which the full compatibility of the history of data associations is checked with the complete data association hypothesis up to the current time. Allowing reconsideration of the established pairings permits one to go back in the past and change the data associations so as to be consistent.

The work of [34] first demonstrated the application of filtering to monocular camera SLAM. In [31], the authors show a filtering based approach to monocular visual odometery that tracks a large number of features in its view with a ransac based outlier rejection technique. Recent work in [66, 92] exploit the use of MEMS-based inertial sensors to fuse high rate odometry data with visual feature tracking to deliver scaled camera motion estimates, this approach is known as Visual-Inertial Odometery (VIO).

In [100], the authors present a particle filtering based SLAM technique that uses Rao-Blackwell factorization. The posterior over history of data is factored into the robot trajectory over the history and the map conditioned on the robot trajectory and measurements. FastSLAM [100] captures the full SLAM problem (history of robot poses and map) online. The algorithm works by maintaining a set of particles where each particle carries its own copy of the map. Secondly, each landmark is estimated with its own Kalman filter, thus, each particle carries a set of Kalman filters, one for each feature. An issue of FastSLAM 1.0 is that it samples the proposal distribution based on the control action and the updates particle weights based on observations. Subsequently in [101], an improvement to FastSLAM is presented which uses actions and observations to calculate the proposal distribution.

4.1.1 SLAM with Relative Feature Measurements

Several works [97, 33, 106] proposed to exploit relative feature measurements in a filtering-based approach. In [33] the correlations between relative measurements from common landmarks are not considered which leads to a sub-optimal estimate. In [106] only relative distances are estimated which neglects the information provided by the direction component of relative measurements. The method of [97] exploits the shift and rotation invariance of map structure but cannot consistently incorporate long range correlations and is thus unable to close loops. In comparison to aforementioned methods [97, 33, 106] our RFM-SLAM formulation in Chapter 6 takes into account both; correlations between relative measurements from common landmarks; and long range correlations between relative measurements in the global frame. This allows RFM-SLAM to form consistent estimates and close large loops. The method of [147] exploits relative feature measurements to decouple map estimation from robot localization in an Extended Information Filter-based formulation, while maintaining

long range correlations. Compared to [147, 97, 33, 106] RFM-SLAM exploits relative measurements to decouple robot orientation estimation from map and robot position. Further RFM-SLAM does not maintain a recursive estimate over the map or robot state, it falls into the category of methods that solve the full SLAM problem.

4.2 Smoothing-based SLAM Methods

The seminal work of [93] introduced the non-linear optimization based approach to solving the full SLAM problem. In this formulation, robot poses are treated as nodes in a graph and constraints as edges. Odometery acts as a weak constraint or edge between 2 successive poses and relative range scan measurements act as a strong edge between all poses that have observation overlap. The method of [17] introduced a generic large-scale mapping methodology that stitches estimates of multiple local maps of restricted size. In [139], the authors unified graph-based approaches and extended it to feature based mapping. In [84], the authors apply large scale graph SLAM to a UAV and incorporate prior information in the form of aerial images. Several other works [37, 82, 48, 94] made important contributions to the original formulation in [93].[1]

A common shortcoming of non-linear optimization based methods is that data has to be processed offline to generate a globally consistent map. To counter this issue several recent works have proposed incremental approaches that exploit the structure of the SLAM problem. The work of [69] developed an incremental approach that exploits sparsity of the information matrix to update only those variables that are affected by incoming data. The work of [72], extends the method of [69] to a graph-based incremental SLAM approach using the method of Bayes trees [71]. In [152] the authors present a real-time lidar based odometry and mapping algorithm that breaks

[1]Several open source implementations can be found online [3], including G2O [88] which is a general purpose non-linear optimization framework for SLAM and iSAM [69].

the problem into a high rate odometery update and slow point cloud registration and bundle adjustment problem. In [149], the authors propose to overcome the non real-time nature of smoothing based methods through a parallel high-speed filtering solution which runs concurrently to large scale loop closure using smoothing and a factor graph approach [83]. The aforementioned method is an extension to [70] which extends the Bayes tree concept to parallel filtering and smoothing. In [103] a multi-state constraint Kalman filter is used in parallel with Bundle Adjustment (BA) for localization.

In computer vision literature, Bundle Adjustment [57] is a common approach for structure from motion that solves for camera and 3D feature poses across multiple images. BA is a non-linear least squares problem for which the Levenberg-Marquardt algorithm [91, 96] is most popular. In [81], a Parallel Tracking and Mapping (PTAM) approach to monocular camera SLAM is presented. Camera pose is tracked at a high rate using a coarse set of features and a larger set of features is mapped in a parallel framework. A keyframe based approach is utilized and BA is used to compute the key frame and map feature poses. PTAM demonstrated significantly better performance and robustness than EKF-based monocular camera SLAM [34] as it can track more features, scales better for larger environments and has the ability to recover in case tracking is lost. Application of PTAM to vision-based autonomous navigation for an aerial vehicle was demonstrated in [45, 126]. Recent work of [43, 44, 27] develop a large scale visual SLAM algorithm that operates on image intensities instead of features and uses filtering to track pose in real-time while using BA to close large loops. In [104], the authors develop a feature-based Visual SLAM method with greater accuracy and robustness than the direct approach of [43]. The method of [104] decouples scene reconstruction from trajectory estimation leading to a significant improvement while highlighting the advantages of features

over featureless approaches in the visual SLAM domain.

The work of [74] captures the problem of connectivity in pose-graph SLAM that affects the accuracy of estimation given pairwise measurements. A simple intuition is that more pathways from a non-anchor node (unknown node) to anchor node (known node) can provide a more reliable estimate. This paper presents a general method to synthesize determinant optimal graphs wherein D-optimality is approximated by a metric named weighted tree connectivity. The authors state that maximizing the D-optimality criterion can be asymptotically achieved by maximizing the weighted tree connectivity (equivalent to maximizing the weighted number of spanning trees) through empirical results in [77].

4.3 Simultaneous Planning Localization and Mapping

In the domain of SPLAM or active SLAM, [47] demonstrated adaptive motion and sensing for feature based-SLAM with underwater sonar data. In [90], a sliding window of past robot poses is maintained in the filter state for mapping partially observable features. In [35], an active approach is taken to visual feature observation by controlling camera pan-tilt. In [135] an active loop closing technique is presented for a grid-based variant of FastSLAM wherein a robot may choose to revisit an older location by tracking a metric and then retraverse the path that was previously followed when that location was visited. In [61] the advantage of multi-step planning horizon strategies was shown as compared to one-step approaches. The method of [131] showed a discretization based planning strategy using the A-optimality cost criterion and demonstrate active loop closure for accurate mapping. In [26] a comparison of uncertainty criteria for EKF-based active SLAM is provided which highlighting advantages of the D-optimality criterion. Several recent works in active SLAM [40, 80, 25, 142, 99, 49] have shown great advances in analyzing and

improving applicability of active SLAM to real systems.

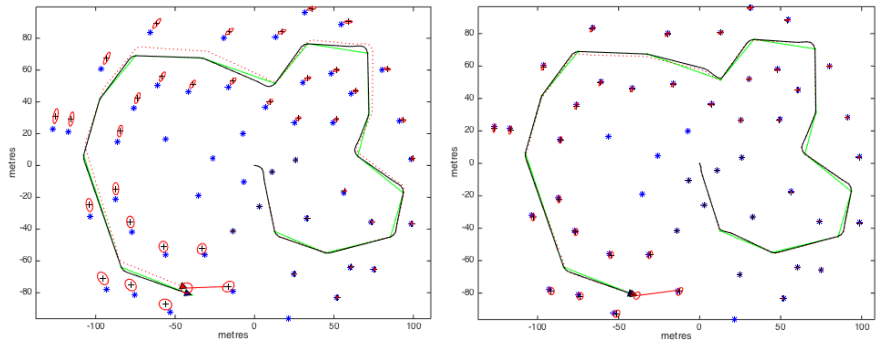
In [127], the authors present a machine learning approach to mapping and planning using long-range vision. A stereo image is captured and then a convolutional neural network classifies the regions of the image into traversable, obstacle or obstacle foot. Regions which are uncertain are given a high cost. The incoming stereo point cloud is mapped into the h-polar (hyperbolic-polar) coordinate system and gridded into cells. The planner is hierarchical/layered in nature. A global Djikstra based solution gives a goal rooted tree which tells the robot where to move for each cell in the map (optimal in terms of traversed distance). A rollout-based local planner is also used which reacts to obstacles locally by finding the local cost to a finite number of candidates and then moves to the candidate that has the lowest sum of cost of to connect and cost-to-go. The global planner runs at 1 Hz and local runs at 5-10 Hz.

Existing SLAM approaches including active SLAM for mobile robot navigation are limited to trajectories $\approx 1-2$km in length with many loop closures that aid in bias correction. Further, most approaches are focused at map building. As opposed to the traditional active SLAM objective of planning for accurate mapping, we approach the problem of planning to map for accurate localization in long trajectories. From a navigation perspective, a sparse high quality map may be better suited for task completion rather than a dense inaccurate map as often there may be a desired accuracy with which the robot should reach its goal. In recent works, the method of [62] approaches this problem. In [62] planning is carried out in the continuous belief space (over robot and feature states) using an RHC-based approach. An uncertainty budget is assigned to the robot and the planner cost formulation penalizes actions that exceed said budget. Local loop closing behavior emerges and the robot tends to turn back to previously visited areas if its pose uncertainty increases.

4.4 The Role of Orientation in SLAM

A key reason for pose and map estimate drift during exploratory motion is the heading estimate of a robot. In the EKF-SLAM formulation, it was shown that heading is unobservable [59] and causes inconsistency in the filter estimates [28, 60, 13] (see Section 4.1). At every filter update, the system model is linearized about the estimated state, due to the highly non-linear nature of these models (trigonometric functions of the heading angle and bearing to features) the estimated Jacobians quickly diverge from the truth. An advantage of pose graph-based methods over filtering is that one can re-linearize the system models (re-compute Jacobians) during iterative refinement in the smoothing process. In essence, one can correct mistakes made in the past. However, both SLAM formulations are non-linear by design and one cannot guarantee a globally optimal solution. Figure 4.2 shows the effect of a direct heading sensor on EKF-based SLAM, Fig. 4.2(b) shows the clear advantage of a heading sensor, this indicates that one can traverse longer into unknown regions without the need for loop closure for the case when there is no heading sensor present.

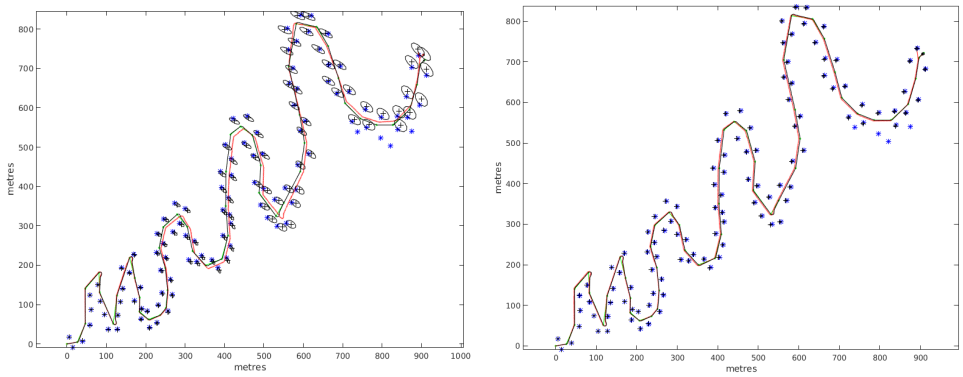
Recent works [98, 24, 22, 79] have sought to exploit structural properties of SLAM with the aim of decoupling non-linearities that arise due to orientation. In [98] the authors develop an incremental SLAM approach that exploits orientation and position separation in 2D range-scan mapping. In [18] a general on-manifold optimization based method is developed to estimate orientations from noisy relative measurements corrupted by outliers. In relation to [18], our orientation estimation method (Section 6.2.2.2) is only concerned with measurement data corrupted by zero-mean Gaussian noise similar to [24, 79]. We direct the reader to [23] for a recent survey of 3D rotation estimation techniques. The works of [24, 22, 79] are closely related to ours, hence we proceed to discuss these in greater detail.



(a) Robot moving at constant velocity of 5m/s and mapping features on the fly.

(b) Robot stops for 60s to make observations when a new feature is observed then moves at 5 m/s.

Figure 4.1: A 2-D EKF-SLAM example ($200m \times 160m$ map) adapted from [14] with no stationary process noise and no heading sensor. In (a) robot moves at a constant velocity, it is seen that filter estimate becomes inconsistent and drift relatively fast (b) stop-go approach allows consistent estimates with slower drift.



(a) When robot has no heading sensor the localization and map quality deteriorates rapidly.

(b) When robot has a heading sensor features are tightly localized and localization accuracy is significantly better.

Figure 4.2: A 2-D EKF-SLAM example where (a) robot does not have a heading sensor and (b)robot has access to a heading sensor.

Linear Approximation for pose Graph Optimization (LAGO) [24] is a method for planar relative pose graph SLAM that separates robot orientation and position estimation into two successive linear problems with the key benefit of a reduced risk

of convergence to local minima and provides a robust initial guess for iterative optimization. The LAGO formulation does not deal with feature-based measurements and cannot be extended to 3D. In contrast, RFM-SLAM is designed for feature-based SLAM and majority of the algorithm presented in this dissertation ports directly to the 3D domain (see discussion in Section **??**). LAGO develops a closed form approach (regularization) to solve the angle wrap-around problem that relies on rounding-off noisy relative orientation measurements. This technique may degrade rapidly once sensor accuracy reduces beyond a certain threshold ([24], Section 6). In contrast, RFM-SLAM does not invoke any such approximation as it computes the maximum likelihood estimate for the orientations via an on-manifold optimization. In this regard, compared to LAGO, our approach trades computational speed, for accuracy and reliability in the orientation estimation phase.

In [22] the authors develop MOLE2D, a multi-hypothesis approach to global orientation estimation from relative measurements that does not suffer from local minima even in the case of high noise. This is achieved through equivalence of the general manifold optimization problem to unconstrained optimization on the integer lattice. Performance guarantees are also provided in the sense that at least one solution is guaranteed to be close to the true underlying global orientation. In relation to our work, MOLE2D provides an alternative to the on-manifold optimization step described in Section 6.2.2.2.

In [79], the authors develop a modified Variable Projection (VP) technique for non-linear optimization that exploits the separation of position and orientation in SLAM and runs faster than the standard Gauss Newton algorithm. The method of [79] solves for orientation and position successively in an iterative manner as opposed to RFM-SLAM wherein iterative non-linear optimization is only applied to orientation estimation. In certain instances the method of [79] may not converge

to a solution ([79], Section 5) as it relies on odometery for the initial guess which may be arbitrarily bad. Our empirical observations indicate that as sensor noise is increased, RFM-SLAM performance degrades gracefully and we do not observe cases where the method does not converge to a solution (see Table 6.1).

4.4.1 Absolute Orientation Sensing for SLAM

A straightforward approach to removing heading estimation biases is to use absolute heading sensors. There exist sensors such as magnetometers, star trackers and gyrocompasses which can provide absolute heading estimates, but each comes with its own host of limitations:

1. Magnetometers: Magnetometers detect the Earth's magnetic field to estimate orientation with respect to the magnetic poles. During indoor operation, magnetometers are affected by fluctuations in the magnetic field due to building materials, electrical wiring etc., thus cannot be trusted to give unbiased heading estimates. During outdoor operation (especially for UAVs), these work relatively better but can be affected by local variations in the magnetic field due to large man-made metallic structures, power cables etc. In recent times MEMS-based magnetometers have become popular due to their small form factor and low-cost ($\approx$ \$10 for a 3 DoF system), these are often combined with MEMS-based inertial sensors to create Attitude Heading Reference Systems (AHRS).

2. Star Trackers: Modern star trackers are automated camera-based devices that compute inertial attitude with high accuracy by using measurements to known celestial bodies and comparing them to star charts (e.g., low-cost systems [4] can easily deliver RMS error down to 10 arcseconds or 0.0028°). Star trackers are extremely robust as they rely on measurements to persistent beacons in

space whose trajectories across the sky relative to Earth or other planets can be fixed with great precision from long-term astronomical observations. Star trackers are used heavily in satellite and space probe navigation [102] as well as on ships and aircraft as a backup navigational source when GPS is unavailable [5, 110] during both day and night operation. The method of using stars to navigate is called celestial navigation.

3. Gyrocompass: A gyrocompass [148] detects the Earth's true north by relying on the concept of gyroscopic precession, thus providing a reliable heading estimate. Gyrocompasses revolutionized marine navigation since their introduction in the 1920s. Commercially available gyrocompasses are heavy, costly and designed primarily for large vehicles, thus this limits their application to commonly available robotics platforms. Further, these sensors have a large settling time and may exhibit minor drift which needs to be corrected with occasional GPS fixes. In recent years, MEMS-based gyrocompasses have been proposed [120] however these are still in prototype stage and far from practical use due to the extremely large settling times ($\approx$ few hours).

In the domain of mobile robotics, celestial navigation has been proposed for planetary rovers as an alternative to existing methods. Current onboard navigation techniques for planetary rovers include fusing inertial measurement units (IMU) with wheel odometery or using visual odometery [95]. In ground-based tests, visual odometery based localization for the Mars Exploration Rover (MER) [95] achieves an accuracy of 2.5% absolute position error over a 24 m course of driving; this may limit how far a rover can traverse accurately during its mission life while relying purely on onboard sensors. In [130] a lunar positioning method is developed that utilizes star tracking and inclinometers however the positioning error is highly sen-

sitive to inclinometer error. Another sensor that aids absolute heading estimation is a sun sensor and has also been proposed for planetary rovers [146, 141]. We note that sun-sensors (error $\approx 1^\circ$ [141]) are not as accurate as star-trackers and do not allow full 3 degree of freedom (DoF) rotation estimation. In [51] an absolute heading estimation technique based on sun-sensing, inclinometer measurements and precision timing is developed for planetary rovers with heading estimation errors of the order of few degrees. Compared to [130, 56, 51, 146, 141] the method proposed in Chapter 5 fuses proprioceptive sensing and exteroceptive sensors such as camera, lidars etc. with star tracking or sun sensors for accurate global navigation. Empirical results show that extremely low localization error (0.0006% of distance traveled) can be easily achieved with star trackers.

4.5 Discussion

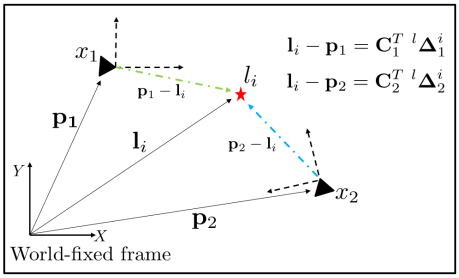
In Chapters 5 and 6 we present 2D feature-based SLAM approaches that exploit the separable structure of the SLAM problem. In Chapter 5 a method for long-term accurate localization is presented for outdoor problems where there may be access to absolute orientation measurements, the primary objective being to analyze the effect of absolute orientation sensing on SLAM. In Chapter 6 we study the problem of 2D feature-based SLAM where absolute orientation measurements may not be available and present a method to separate orientation and position estimation. Finally in Chapter 7 we apply the concept of absolute orientation sensing to indoor 2D LiDAR-based SLAM in a physical real-world setting.

5. LINEAR FEATURE GRAPH OPTIMIZATION

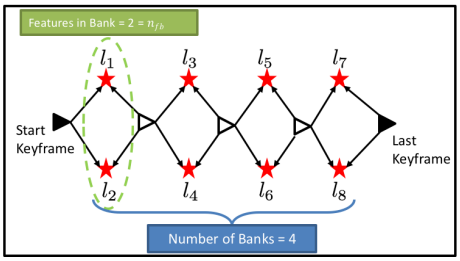
In SLAM, estimation errors tend to build up during exploratory motion and are usually negated by revisiting previously seen locations (loop closure). Certain mobile robotics applications, e.g., autonomous military vehicles and planetary exploration may require robots to accurately traverse long trajectories (long-term point-to-point navigation) without revisiting previous locations and wherein global localization information (e.g., GPS) may be unavailable or degraded. State-of-the-art Visual-Inertial localization methods [65, 92] exhibit error of $\approx 0.3\% - 0.5\%$ which may be unsuitable for precision tasks, e.g. for a 25 km trajectory, it results in $75\text{ m} - 125\text{ m}$ position error.

A key reason for estimation drift during exploration is the robot heading uncertainty. In common practice, it is assumed that reliable absolute orientation measurements are *not* available in SLAM. Thus existing methods rely on odometery and relative pose or feature measurements to estimate robot orientation and position. We now proceed to explore the role of absolute orientation sensing in SLAM and its impact on point-to-point autonomous navigation. We ask the question, "Is there a way for a robot to sense its orientation robustly without relying on odometery or local relative measurements?", the answer as it turns out is *yes*. In Section 5.3 we show that it is possible to attain an accuracy of 0.0006% for a 107.9 km trajectory without loops using existing absolute orientation sensor technology. In fact, such sensors are being mandated for GPS-denied navigation on military aircraft and ships [2, 110].

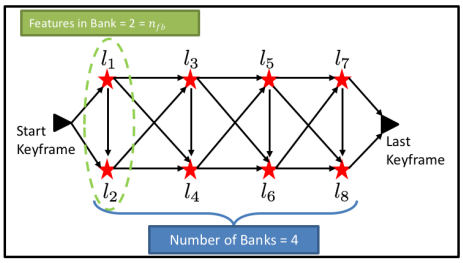
In Section 5.1 we state the problem definition and subsequently, in Section 5.2 we present our approach. In Section 5.3 we present simulation results for a robot that visits a pre-determined sequence of waypoints followed by conclusions in Section 5.4.



(a) A robot sees landmark l_i from two poses x_1, x_2 and estimates the relative positions ${}^l\boldsymbol{\Delta}_1^i$, ${}^l\boldsymbol{\Delta}_2^i$ of l_i in its local frames.



(b) R2F: Robot makes robot to feature relative position measurements.



(c) F2F: Robot makes relative feature to feature displacement measurements.

Figure 5.1: Hypothetical scenarios depicting robot operation; (a) robot transforms local measurements to the global frame using orientation sensing; In (b) and (c) robot observes four banks of features prior to final pose with two features in each bank and uses approach R2F and F2F in (b) and (c) respectively.

5.1 Problem Formulation

Let x_k, u_k, and z_k represent the system state, control input, and observation at time step k respectively. The state vector $\mathbf{x}_k = [\mathbf{p}_k^T, \boldsymbol{\theta}_k^T]^T$, where $\mathbf{p}_k$ and $\boldsymbol{\theta}_k$ are the robot position and orientation parameter vector respectively. We designate a keyframe pose as ${}^{\kappa}x$. The state evolution model f is denoted as $x_{k+1} = f(x_k, u_k) + w_k$ where $w_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ is zero-mean Gaussian process noise. The measurement model h is denoted as $z_k = h(x_k) + v_k$, where $v_k \sim \mathcal{N}(0, \mathbf{R}_k)$ is zero-mean Gaussian measurement noise. The map (unknown at t_0) is a set of landmarks (features) distributed throughout the environment. We define the j-th landmark as l_j. The observation for landmark l_j at time t_k is denoted by $z_k^j \in z_k$. The non-linear inverse measurement model is denoted by g such that for a given measurement z_k^j and the state x_k at which it was made, g computes the landmark location $l_j = g(x_k, z_k^j)$.

We define ${}^{l}\mathbf{d}_k^{ij}$ to be the relative feature measurement, i.e., displacement from feature l_i to l_j in robot's local frame at time t_k. The local relative feature measurement ${}^{l}\mathbf{d}_k^{ij} = {}^{l}\boldsymbol{\Delta}_k^j - {}^{l}\boldsymbol{\Delta}_k^i$, where ${}^{l}\boldsymbol{\Delta}_k^i, {}^{l}\boldsymbol{\Delta}_k^j$ are relative positions of features l_i and l_j respectively with respect to the robot in its local frame. Let $\mathbf{C}(\boldsymbol{\theta}_k)$ denote the Direction Cosine Matrix (DCM) of the robot orientation at state x_k where $\mathbf{C}$ is a function of orientation parameter $\boldsymbol{\theta}_k$ (e.g., Euler angles, Quaternions etc.). A measurement ${}^{l}\boldsymbol{\Delta}_k^i$ in the robot's local frame can be projected into the world (global) frame as

$$\mathbf{l}_i - \mathbf{p}_k = \underbrace{{}^{w}\boldsymbol{\Delta}_k^i}_{\text{world frame measurement}} = \mathbf{C}(\boldsymbol{\theta}_k)^T \, {}^{l}\boldsymbol{\Delta}_k^i, \tag{5.1}$$

where $\mathbf{l}_i$ and $\mathbf{p}_k$ are the feature and robot positions in the world frame, see Fig. 5.1(a). Currently, the focus of this work is on analyzing the effect of orientation, thus we do not model obstacles in simulation studies. Data associations to landmarks are assumed given and map is considered static.

5.2 Methodology

In this section we develop two approaches for long-term localization that differ in the way measurements are processed:

1. **Robot to Feature Relative Position Measurement Model (R2F)**: An approach designed for systems where a robot moves continuously and relies on odometery, orientation sensing and exteroceptive sensing like Lidar, cameras etc. This method converts local relative position measurements of features w.r.t robot at each pose to global frame measurements. These are then used to solve a linearized least squares estimation problem in robot and feature positions. Figure 5.1(b) depicts this approach.

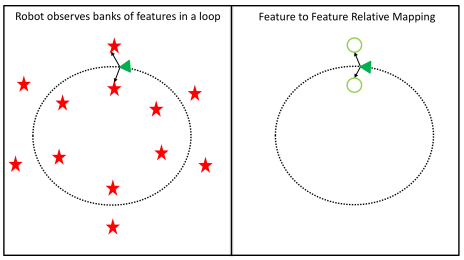
2. **Feature to Feature Relative Measurement Model (F2F)**: An approach designed for systems where odometery may not provide reliable information but time budgets may be higher. In this approach, independent measurements are made for relative displacements between features which are then chained together to estimate the map and robot position. Figure 5.1(c) depicts this approach.

5.2.1 R2F

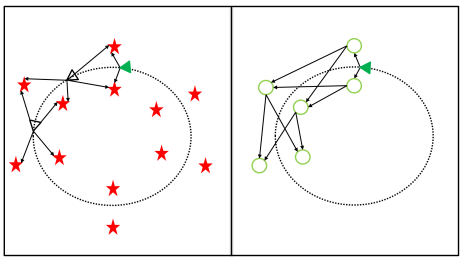
The key steps of this approach are as follows:

1. Range bearing measurements to features are transformed to relative position estimates of features w.r.t robot at each pose (see Fig. 5.1(a)).

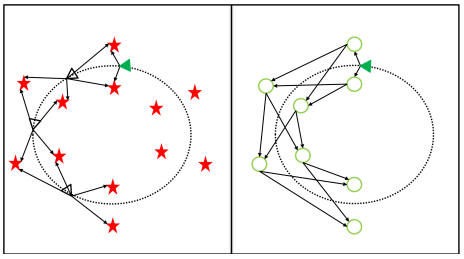
2. At keyframe poses, a linearized estimation problem is solved for the robot pose and features using the recorded data.



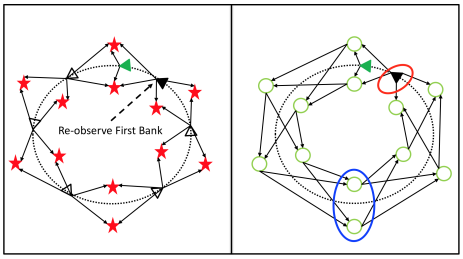
(a) Robot starts making observations to first bank of features.



(b) Robot makes relative observations between second and third bank of features.



(c) Robot is moving towards its start location.



(d) Robot re-observes the first bank of features.

Figure 5.2: A simple graphical depiction of loop closure, the left half of each image shows the robot making range bearing measurements and the right half shows feature to feature mapping. In Fig. 5.2(d) we are interested in the estimation error of the farthest feature bank (encircled by blue ellipse) and the last pose (encircled by red ellipse).

3. Once the estimation problem is solved, only correlations between keyframe poses and features observed at keyframe poses are maintained, intermediate feature and pose estimates are dropped.

4. An upper threshold is set on the number of keyframes to keep in the map after which the oldest keyframe is deleted (excluding first pose).

We now proceed to explain this approach in detail. Using the inverse measurement model g (Section 5.1), we have the position of landmark l_i in robot's local frame as ${}^{l}\mathbf{\Delta}_k^i = {}^{l}\mathbf{g}(\mathbf{z}_k^i)$, extending this, the vector of local robot to feature relative position measurements is ${}^{l}\hat{\mathbf{\Delta}}_k = {}^{l}\mathbf{g}(\mathbf{z}_k)$. Thus ${}^{l}\hat{\mathbf{\Delta}}_k \sim \mathcal{N}({}^{l}\mathbf{\Delta}_k, {}^{l}\mathbf{R}_{\mathbf{\Delta}_k} = \bar{\nabla}{}^{l}\mathbf{g}|_{\mathbf{z}_k}\mathbf{R}_{\mathbf{z}_k}\bar{\nabla}{}^{l}\mathbf{g}^T|_{\mathbf{z}_k})$, where $\bar{\nabla}{}^{l}\mathbf{g}|_{\mathbf{z}_k}$ is the Jacobian of function ${}^{l}\mathbf{g}(\mathbf{z}_k)$. At t_0, we have keyframe ${}^{\kappa}x_0$, the robot starts moving and collecting odometery, orientation and exteroceptive measurements at each timestep. At some future timestep t_k, we have keyframe ${}^{\kappa}x_k$. Abusing notation slightly, let ${}^{l}\hat{\mathbf{\Delta}}_{0:k} \sim \mathcal{N}({}^{l}\mathbf{\Delta}_{0:k}, {}^{l}\mathbf{R}_{\mathbf{\Delta}_{0:k}})$ be the vector of all local relative position measurements (including robot to feature and translational odometery) recorded at poses $x_{0:k}$. At each t_k we have a noisy unbiased orientation measurement which gives us the vector $\hat{\boldsymbol{\theta}}_{0:k} \sim \mathcal{N}(\boldsymbol{\theta}_{0:k}, \mathbf{R}_{\boldsymbol{\theta}_{0:k}})$. Dropping the time subscript for readability, local relative measurements ${}^{l}\hat{\mathbf{\Delta}}$ can be transformed to the world frame as

$$ {}^{w}\hat{\mathbf{\Delta}} = \mathbf{C}^T(\hat{\boldsymbol{\theta}})\ {}^{l}\hat{\mathbf{\Delta}}, \tag{5.2} $$

where $\hat{\mathbf{C}} = \mathbf{C}(\hat{\boldsymbol{\theta}})$ is the corresponding composition of DCM matrices. From Eq. 5.2 it is clear that transformed global measurements at each pose $\mathbf{x}_k$ are correlated to heading measurement $\hat{\boldsymbol{\theta}}_k$. Heading error covariances must be propagated appropriately in the feature and robot position estimation. We setup a new measurement model $\boldsymbol{\beta} = \mathbf{h}_\beta({}^{l}\mathbf{\Delta}, \boldsymbol{\theta}) + \mathbf{v}_{\boldsymbol{\beta}}$ by stacking the transformed odometery between poses,

robot to feature displacement and heading measurements. This gives us the following problem

$$\boldsymbol{\beta} = \begin{bmatrix} \hat{\mathbf{C}}^T \, {}^l\hat{\boldsymbol{\Delta}} \\ \hat{\boldsymbol{\theta}} \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} \mathbf{A}' & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right]}_{\mathbf{A}} \left[\begin{array}{c} \mathbf{p} \\ \mathbf{l} \\ \hline \boldsymbol{\theta} \end{array}\right] + \mathbf{v}_{\boldsymbol{\beta}}, \tag{5.3}$$

where $[\mathbf{p}^T, \mathbf{l}^T, \boldsymbol{\theta}^T]^T$ is the vector of robot poses, landmarks and heading, $\mathbf{A}'$ is a matrix with each row containing elements of the set $\{-1, 0, +1\}$, $\mathbf{v}_{\boldsymbol{\beta}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{\boldsymbol{\beta}})$ is the measurement noise vector with covariance

$$\mathbf{R}_{\boldsymbol{\beta}} = \bar{\nabla}\mathbf{h}_\beta \begin{bmatrix} {}^l\mathbf{R}_{\boldsymbol{\Delta}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \bar{\nabla}^T\mathbf{h}_\beta, \tag{5.4}$$

where $\bar{\nabla}\mathbf{h}_\beta$ is the Jacobian of measurement function $\mathbf{h}_\beta$ given by

$$\bar{\nabla}\mathbf{h}_\beta = \begin{bmatrix} \mathbf{C}^T & \mathbf{M} \, {}^l\hat{\boldsymbol{\Delta}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \; \mathbf{M} = \frac{\partial \hat{\mathbf{C}}^T}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}. \tag{5.5}$$

The solution to the problem in Eq. 5.3 is given by

$$\begin{bmatrix} \mathbf{p}^* \\ \mathbf{l}^* \\ \boldsymbol{\theta}^* \end{bmatrix} = (\mathbf{A}^T\mathbf{R}_{\boldsymbol{\beta}}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{R}_{\boldsymbol{\beta}}^{-1}\boldsymbol{\beta}. \tag{5.6}$$

From Eq. 5.6 we are interested in $\mathbf{p}^*$ and $\mathbf{l}^*$.

5.2.1.1 Analysis

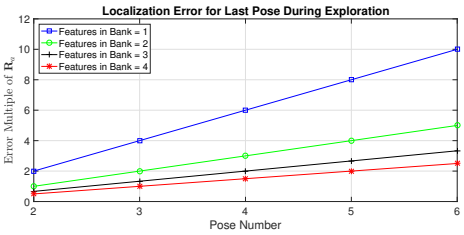
We analyze localization accuracy for approach R2F as a robot explores an unknown map and the effect of loop closure. We assume that relative measurements

from robot to features in the global frame are independent; for orientations sensors such as star trackers which are highly precise such an assumption holds quite well. For the sake of clarity error covariance of every global relative measurement is assumed to be $\mathbf{R}_a$. In this analysis, the first and last pose of the robot are considered as keyframes with first pose known.

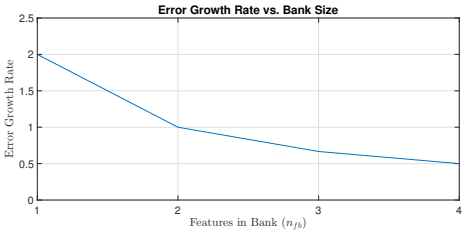
Exploration: In an exploration task, a robot moves into unknown areas and makes measurements to a bank of features at each timestep as shown in Fig. 5.1(b). A bank is simply the set of features observed at a particular pose. Figure 5.3(a) shows that the error growth is linear as robot moves further away from its initial pose. An interesting point arises in Fig. 5.3(b), the rate of error growth drops as $\propto 1/n_{fb}$ where n_{fb} is the number of features in each bank. Thus it implies that for long-term autonomy, it benefits us to make high-quality observations to a large number of features but we may not always need hundreds or thousands of features. Rather one may predict localization error for a given task and decide how many features to observe. This can help in constraining computational complexity and memory resources.

Loop Closure: Loop closure is usually considered a necessity in SLAM to limit error growth. We consider the effect of loop closure on the bank of features farthest from the start location in a loop and on the last pose (see Fig. 5.2):

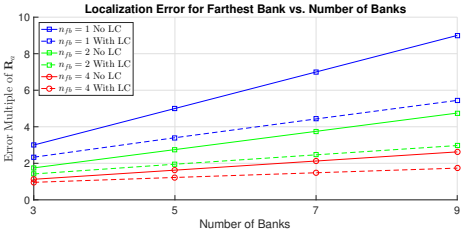
1. For the error in farthest bank (encircled by a blue ellipse in Fig. 5.2(d)), Fig. 5.3(c) shows that the effect of loop closure diminishes as the bank size increases (slope of solid vs. dashed lines). Note that the ratio of error growth rate after and before loop closure is 0.5185, i.e., loop closure approximately halves the error growth in the farthest bank.



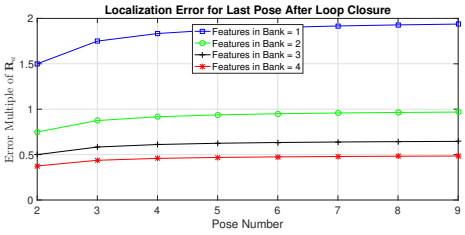
(a) The error growth in estimate of final pose as a function of how far the robot moves.



(b) The position estimation error growth rate drops $\propto 1/n_{fb}$ as the number of landmarks in each bank n_{fb} increases.



(c) Localization error in the feature bank farthest from start before (solid lines) and after (dashed lines) loop closure.



(d) Localization error in the last pose after loop closure as the trajectory length (number of banks mapped) increases.

Figure 5.3: **R2F**: Analysis of the pose and map estimation error as the robot explores unknown regions and the effect of loop closure when robot re-observes features mapped at first pose.

2. Figure 5.3(d) shows that error in last pose after loop closure converges to a fixed value as the trajectory length increases.

The first observation noted above can be understood intuitively. Looking at Fig. 5.2(d), there are two "pathways" that link the farthest bank of features to the first bank. Prior to loop closure there is only one path for the relative measurements to constrain feature estimates to the first bank, however, after loop closure there is a second pathway from the opposite direction. If we were to take two observations ($\mathbf{R}_a$ is halved) for each feature, we would effectively end up with the same estimation error at the farthest bank were the robot not to close the loop. The second observation tells us that the estimation error in the last bank of features is dominated by the relative measurement to the first bank, i.e., as the trajectory length grows, the effect of the longer "pathway" from start has almost no effect on estimation accuracy.

5.2.2 F2F

The key steps in approach F2F are as follows:

1. As the robot moves, range bearing measurements to features are transformed into relative displacement estimates between features.

2. Robot to feature relative position measurements are acquired at keyframes. Then a linearized estimation problem is solved for keyframe poses and map features using the recorded data.

3. Once the estimation problem is solved, only correlations between keyframe poses and features observed at keyframes are maintained, intermediate feature estimates are dropped.

4. An upper threshold is set on the number of keyframes to keep in the map after which the oldest keyframe is deleted (excluding first pose).

We now proceed to explain this approach in detail. At time t_k we have the measurement of displacement vector from l_i to l_j in the robot frame as

$$ {}^{l}\mathbf{d}_k^{ij} = {}^{l}\boldsymbol{\Delta}_k^{j} - {}^{l}\boldsymbol{\Delta}_k^{i} = {}^{l}\mathbf{g}_d(\mathbf{z}_k^j, \mathbf{z}_k^i). \tag{5.7} $$

The above equation shows that ${}^{l}\mathbf{d}_k^{ij}$ is independent of robot position $\mathbf{p}_k$ and orientation $\boldsymbol{\theta}_k$. The vector of local relative measurements is ${}^{l}\hat{\mathbf{d}}_k = {}^{l}\mathbf{g}_d(\mathbf{z}_k)$. Note that though range bearing measurements to each feature are independent, the set of relative feature measurements may be *correlated* in a naive implementation due to the correlations between relative measurements using the same range-bearing observation. This is where a key difference from existing methods arises; our method takes independent relative measurements by capturing sufficient range bearing observations to features.

Let ${}^{l}\hat{\mathbf{d}}_k \sim \mathcal{N}({}^{l}\mathbf{d}_k, {}^{l}\mathbf{R}_{\mathbf{d}_k} = \bar{\nabla}{}^{l}\mathbf{g}_d|_{\mathbf{z}_k}\mathbf{R}_{\mathbf{z}_k}\bar{\nabla}{}^{l}\mathbf{g}_d^T|_{\mathbf{z}_k})$ be the vector of independent relative feature measurements at time t_k with error covariance ${}^{l}\mathbf{R}_{\mathbf{d}_k}$, where $\bar{\nabla}{}^{l}\mathbf{g}_d|_{\mathbf{z}_k}$ is the Jacobian of the local relative measurement function ${}^{l}\mathbf{g}_d(\mathbf{z}_k)$. The robot collects orientation and exteroceptive measurements at each timestep. At some future timestep t_k, we have keyframe ${}^{\kappa}x_k$. Dropping the time subscript for legibility. Let ${}^{l}\hat{\mathbf{d}} \sim \mathcal{N}({}^{l}\mathbf{d}, {}^{l}\mathbf{R}_{\mathbf{d}})$ be the vector of local relative feature to feature measurements captured from time t_1 to t_k. At keyframes ${}^{\kappa}x_0$ and ${}^{\kappa}x_k$ we have robot to feature relative measurements which give us ${}^{l}\hat{\boldsymbol{\Delta}} \sim \mathcal{N}({}^{l}\boldsymbol{\Delta}, {}^{l}\mathbf{R}_{\boldsymbol{\Delta}})$. At each pose we have a noisy unbiased heading measurement which provides the vector of orientation estimates $\hat{\boldsymbol{\theta}} \sim \mathcal{N}(\boldsymbol{\theta}, \mathbf{R}_{\boldsymbol{\theta}})$. The vectors of local relative measurements ${}^{l}\hat{\boldsymbol{\Delta}}, {}^{l}\hat{\mathbf{d}}$ can be transformed to the world frame as

$$\begin{bmatrix} {}^{w}\hat{\mathbf{d}} \\ {}^{w}\hat{\boldsymbol{\Delta}} \end{bmatrix} = \hat{\mathbf{C}}^{T} \begin{bmatrix} {}^{l}\hat{\mathbf{d}} \\ {}^{l}\hat{\boldsymbol{\Delta}} \end{bmatrix}. \tag{5.8}$$

Heading error covariances must be propagated appropriately in the feature and robot position estimation. Thus we setup a new measurement model $\boldsymbol{\gamma} = \mathbf{h}_{\gamma}({}^{l}\boldsymbol{\Delta}, {}^{l}\mathbf{d}, \boldsymbol{\theta}) + \mathbf{v}_{\boldsymbol{\gamma}}$ by stacking the transformed relative position and heading measurements. This gives us the following problem

$$\boldsymbol{\gamma} = \begin{bmatrix} \hat{\mathbf{C}}^{T}\ {}^{l}\hat{\mathbf{d}} \\ \hat{\mathbf{C}}^{T}\ {}^{l}\hat{\boldsymbol{\Delta}} \\ \hat{\boldsymbol{\theta}} \end{bmatrix} = \underbrace{\left[\begin{array}{c|c} \mathbf{A}' & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right]}_{\mathbf{A}} \left[\begin{array}{c} \mathbf{l} \\ \mathbf{p} \\ \hline \boldsymbol{\theta} \end{array}\right] + \mathbf{v}_{\boldsymbol{\gamma}}, \tag{5.9}$$

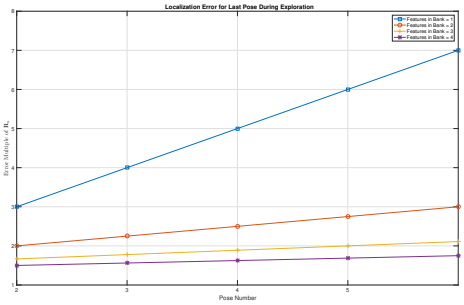
where $[\mathbf{l}^{T}, \mathbf{p}^{T}, \boldsymbol{\theta}^{T}]^{T}$ is the vector of landmark positions, robot poses and heading, $\mathbf{A}'$ is a matrix with each row containing elements of the set $\{-1, 0, +1\}$, $\mathbf{v}_{\gamma} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{\boldsymbol{\gamma}})$ is the measurement noise vector where

$$\mathbf{R}_{\boldsymbol{\gamma}} = \bar{\nabla}\mathbf{h}_{\gamma} \begin{bmatrix} {}^{l}\mathbf{R}_{\mathbf{d}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & {}^{l}\mathbf{R}_{\boldsymbol{\Delta}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \bar{\nabla}^{T}\mathbf{h}_{\gamma}. \tag{5.10}$$

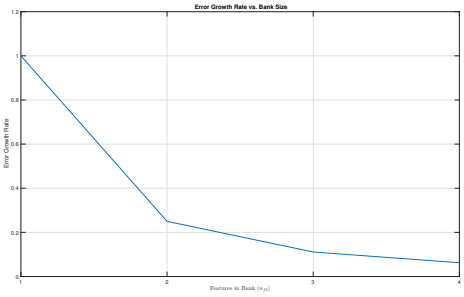
$\bar{\nabla}\mathbf{h}_{\gamma}$ is the Jacobian of measurement function $\mathbf{h}_{\gamma}$ given by

$$\bar{\nabla}\mathbf{h}_{\gamma} = \begin{bmatrix} \mathbf{0} & \mathbf{C}^{T} & \mathbf{M}\ {}^{l}\hat{\mathbf{d}} \\ \mathbf{C}^{T} & \mathbf{0} & \mathbf{M}\ {}^{l}\hat{\boldsymbol{\Delta}} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}, \text{where } \mathbf{M} = \frac{\partial \hat{\mathbf{C}}^{T}}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}. \tag{5.11}$$

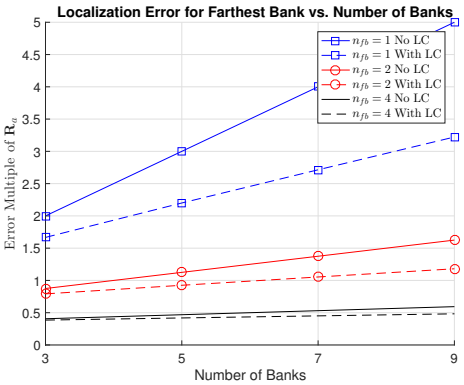
The solution to the problem in Eq. 5.9 is given by



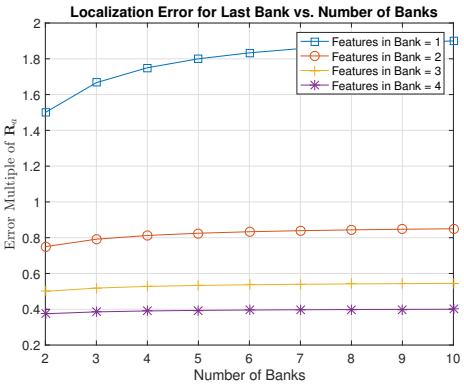
(a) The error growth in estimate of the last pose as a function of how far the robot moves. The growth is linear as robot moves away from its start.



(b) The error growth rate drops sharply $\propto 1/n_{fb}^2$ as the number of landmarks in each bank n_{fb} increases.



(c) Localization error in the bank farthest from start before (solid lines) and after (dashed lines) loop closure.



(d) Localization error in the last bank after loop closure as the trajectory length (number of banks mapped) increases.

Figure 5.4: **F2F**: Analysis of the feature mapping and localization error as the robot moves and the effect of loop closure.

$$\begin{bmatrix} \mathbf{l}^* \\ \mathbf{p}^* \\ \boldsymbol{\theta}^* \end{bmatrix} = (\mathbf{A}^T \mathbf{R}_{\boldsymbol{\gamma}}^{-1} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{R}_{\boldsymbol{\gamma}}^{-1} \boldsymbol{\gamma}. \tag{5.12}$$

5.2.2.1 Analysis

We proceed to analyze how the localization error grows as a robot explores an unknown map and the affect of loop closure with approach F2F. By virtue of the reasoning provided in Section 5.2.1.1, it is assumed that the global frame relative feature measurements are independent and error covariance of every global relative measurement is $\mathbf{R}_a$. Again, the first and last pose of the robot are considered as keyframes with the first pose known.

Exploration: We analytically solve the estimation problem of Eq. 5.9 and compute error covariance for multiple cases by varying bank size n_{fb} and the number of banks that the robot maps. Figure 5.4 shows our results, the plots highlight two key points:

1. Figure 5.4(a) shows that localization error grows linearly as the robot moves away from the start location.

2. The error growth rate shown in Fig. 5.4(b) is inversely proportional to the square of the size of each feature bank, i.e., $\propto \frac{1}{n_{fb}^2}$ where n_{fb} is the number features in one bank. Thus error growth with F2F is $\frac{1}{n_{fb}}$-th of the error growth with R2F.

Loop Closure: Figure 5.2 depicts loop closure when a robot makes relative feature measurements while moving in a circular trajectory. Our goal is to understand the effect of loop closure on mapping accuracy, particularly error in estimation of farthest feature bank from start and last keyframe at which robot re-observes landmarks seen

at its starting location (see Fig. 5.2(d)). We solve Eq. 5.9 with loop closure. The results of this analysis are plotted in Figs. 5.4(c) and 5.4(d); the key points to be noted are:

1. The ratio of error growth rate with loop closure to error growth rate without loop closure is a constant value of 0.5181 for all values of n_{fb} computed as the ratio of slopes of curves plotted in Fig. 5.4(c) before and after loop closure.

2. Figure 5.4(d) shows that error in last pose after loop closure converges to a fixed value as the trajectory length (number of banks) increases.

Thus the behavior of loop closure with method F2F is similar to that with R2F, i.e., loop closure halves the estimation error of farthest feature bank and error in last pose estimate is dominated by measurement to the first bank of features.

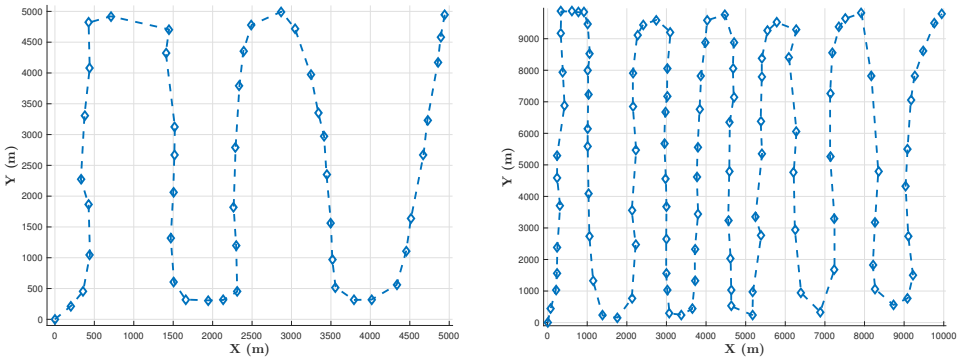
5.3 Results

Simulation results are presented for waypoint following in a 2D environment. These simulations study the case of a long-term exploration task where a robot may not visit prior locations. Robot kinematics are modeled as a steered bicycle [13]. The robot moves at a speed of 10 m/s and simulation time step is 0.05 s. Keyframes are recorded every 100 time steps or 5 s, and a maximum of 4 keyframes are kept in memory. The robot is equipped with three sensors; star tracker with noise standard deviation $\sigma_\theta = 0.005^\circ$ (18 arcseconds); range bearing sensor with a 360° field-of-view, range of 20 m and noise standard deviation of $\sigma_r = 0.1$ m in range and $\sigma_\phi = 1.0^\circ$ in bearing; wheel odometery noise is scaled according to factor $\alpha = \{1, 2, 3, 4\}$ with $\alpha = 4$ corresponding to noise standard deviation $\sigma_{\delta x, \delta y} = 0.1$ m in translation and $\sigma_{\delta\theta} = 1.2^\circ$ in rotation. Two different scenarios are presented:

1. **Scenario A**: A 2D world (5 km $\times$ 5 km) with a trajectory of length 25.9 km as depicted in Fig. 5.5(a).

2. **Scenario B**: A 2D world (10 km $\times$ 10 km) with a trajectory of length 107.9 km as depicted in Fig. 5.5(b).

For each scenario, multiple versions of the environments were generated by varying n_{fb}, i.e., the minimum number of features visible at each pose and then randomly sampling feature locations. Simulations results for a robot equipped only with wheel odometery and orientation sensing are plotted in Fig. 5.6 for both scenarios. In simulations for method R2F, only the highest odometery noise level, i.e., $\alpha = 4$ is considered, further recall that method F2F does not require odometery. Finally, results for approaches R2F and F2F are plotted in Figures 5.7 and 5.8 for scenarios A and B respectively. In total, 1200 simulations were conducted.



(a) Scenario A with 25.9 km trajectory.

(b) Scenario B with 107.9 km trajectory.

Figure 5.5: The robot is tasked to follow the waypoints (depicted by diamonds) sequentially. Note that the trajectory terminates far from the start location and there are no loop closures in the trajectory by design.

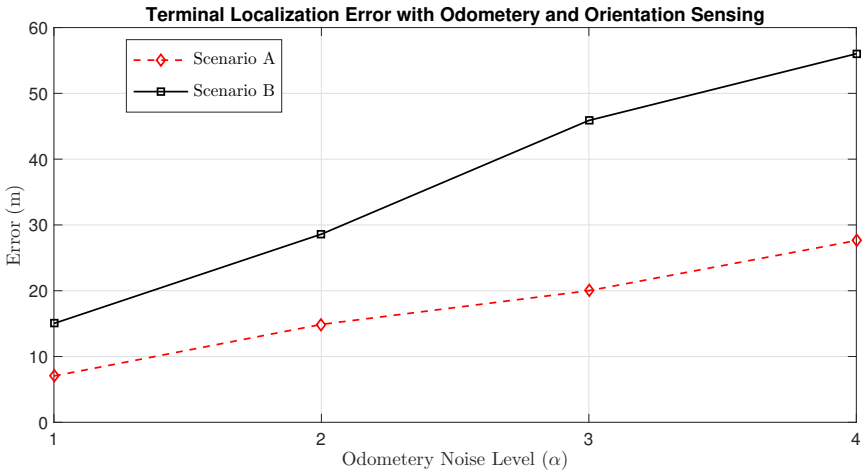
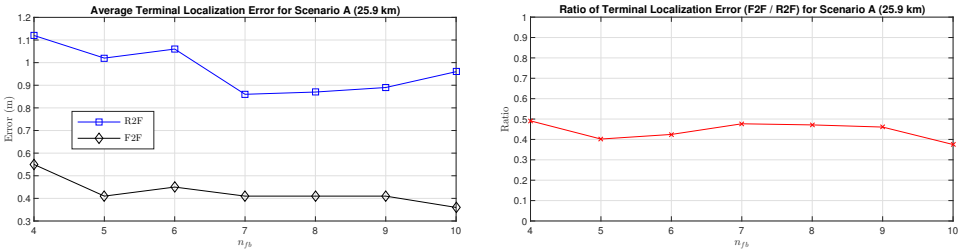


Figure 5.6: The average terminal localization error in Scenarios A and B when robot uses an absolute orientation sensor coupled with odometery.

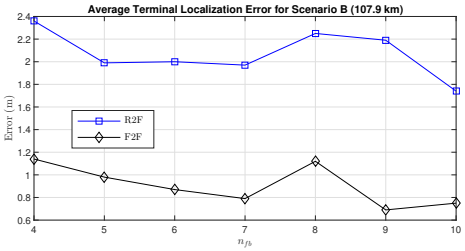


(a) Average terminal localization error for proposed approaches.

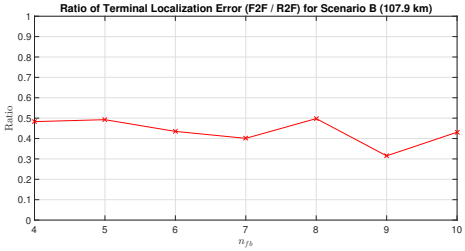
(b) Ratio of error with approach F2F to approach R2F.

Figure 5.7: Scenario A: (a) Average terminal localization error; and (b) ratio of localization error with F2F to error with R2F as the minimum number of features (n_{fb}) visible at each pose is varied. R2F results are depicted in blue with square markers and F2F in black with diamond markers.

For a robot equipped only with odometery and orientation sensing (Fig. 5.6) with $\alpha = 4$, in scenario A error is 27.63 m and in scenario B error is 56 m. Worst case error ($n_{fb} = 4$) in scenario A for method R2F (1.12 m) and F2F (0.55 m) and for scenario B with method R2F (2.36 m) and F2F (1.14 m) are on average $1 - 2$ orders of magnitude lower. Note that odometery noise at 1σ corresponds to



(a) Average terminal localization error for the two proposed approaches.



(b) Ratio of error with approach F2F to approach R2F.

Figure 5.8: Scenario B: (a) Average terminal localization error; and (b) ratio of localization error with F2F to error with R2F as the minimum number of features (n_{fb}) visible at each pose is varied.

20% error in relative position estimate for motion in a single time step which is relatively high, however in practical applications wheel slip may go as high as 125% [95] (steep gradients, loose gravel or soil may cause wheel spin in-place). While using odometery with orientation performs reasonably, it may not meet requirements for certain missions, be reliable or robust in practical applications and may not scale well with longer trajectories. Using measurements to features on the other hand betters performance while making our method robust and allows control over the error growth rate.

Figure 5.7(a) shows the reduction in localization error at the final waypoint in scenario A (25.9 km), for approach R2F and F2F. R2F results in a 23.2% error reduction from 1.12 m at $n_{fb} = 4$ to 0.86 m at $n_{fb} = 7$. There is a minor 0.1 m increase in error from $n_{fb} = 7$ to $n_{fb} = 10$ which can be attributed to the random placement of landmarks; recall that in the physical world error in relative position measurements to features is governed by the range and bearing to a feature, i.e., a far feature's relative position estimate w.r.t robot is more sensitive to error in bearing measurement than a closer feature. With F2F error reduces by 34.5% from 0.55 m

at $n_{fb} = 4$ to 0.36 m at $n_{fb} = 10$. Figure 5.7(b) shows that F2F results in $40 - 50\%$ error relative to R2F, recall that F2F does not incorporate odometery. This supports findings of MER missions [95] wherein wheel slip on high-gradient slopes require the rover to rely on visual odometery to estimate motion. Figure 5.8 depicts localization error at the final waypoint in scenario B (107.9 km), for approach R2F and F2F. R2F results in a 26.3% error reduction from 2.36 m at $n_{fb} = 4$ to 1.74 m at $n_{fb} = 10$. F2F results in a 29.6% reduction in error from 0.98 m at $n_{fb} = 4$ to 0.69 m at $n_{fb} = 9$. Figure 5.8(b) shows that F2F results in $40 - 50\%$ error relative to R2F similar to scenario A.

Compared to existing methods that rely on absolute orientation sensing, our approach results in order(s) of magnitude better performance. The method of [56] fuses inertial sensing with star tracking for autonomous navigation and demonstrated a localization error of ≈ 2 m for a ≈ 3.8 km trajectory[1]. In the lunar positioning method of [130], for the case of tilt estimation error in both roll and pitch axes of 0.05°, star tracker boresight accuracy of 0.0014°, and boresight roll of 0.01°, the resulting error is 1.2 km. The approach of [146] using a sun sensor fused with inertial data and wheel odometery for navigation resulted in a 6% cross-track position error for 1 km of driving.

5.4 Discussion

Two approaches for feature-based localization and mapping were developed for the case when global orientation sensing is available. In method R2F, localization error growth rate is $\propto 1/n_{fb}$, and for method F2F error growth rate is $\propto 1/n_{fb}^2$, where n_{fb} is the minimum number of features visible to a robot at each time. Analysis for both methods shows that; (i) given unbiased heading measurements, localization

[1]Numerical values ascertained from graphical plots in [56] to the best of our understanding.

error growth is linear as robot moves away from its start location; and (ii) loop closure results in $\approx 50\%$ reduction in localization error of features farthest from loop start location. Analysis also indicates that estimation accuracy may be enhanced by taking prolonged measurements which is more suitable for a point-to-point navigation task, thus avoiding loop closure when absolute heading is available. Simulations show that using the proposed approaches order(s) of magnitude better performance is achievable, i.e., with as low as 9 features being tracked at any time, a robot can achieve a position accuracy of 0.69 m for > 100 km of driving.

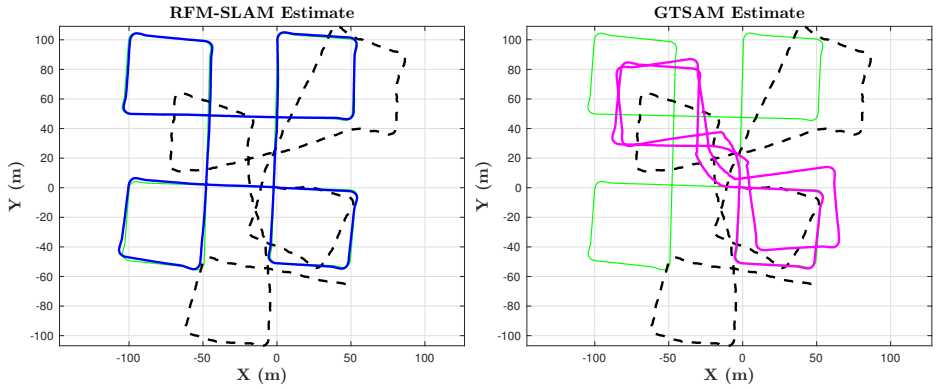
Most robots deployed in physical applications today do not have access to absolute orientation sensors. Thus it is important to the study the problem of feature-based SLAM when heading may not be available. In the next Chapter, we look at the problem of 2D feature-based SLAM where heading measurements may not be available and show how exploiting the separable structure of the underlying SLAM problem allows for significantly more robust estimation compared to state-of-the-art methods.

6. RFM-SLAM: EXPLOITING RELATIVE FEATURE MEASUREMENTS TO SEPARATE ORIENTATION AND POSITION ESTIMATION IN SLAM

Relative measurements [106, 33, 128, 97, 118] allow a robot to exploit structural properties of the environment, e.g., relative displacement from one landmark to another is independent of how a robot moves in a static world given a particular frame of reference. Taking note of this property, we present a 2D SLAM approach in which range bearing measurements are transformed into relative displacements between features. In our method, relative orientation constraints between poses are formulated using translation and rotation invariant structural properties. This allows our method to exploit the separable structure of SLAM [79, 24, 22], i.e., robot heading estimation is separated from the estimation of past robot positions and feature locations. Using relative orientations between the set of robot poses, our method solves a non-linear optimization problem over the set of robot orientations following which we solve a linear least squares problem for position (robot trajectory and map). We call this method Relative Feature Measurements-based Simultaneous Localization and Mapping (RFM-SLAM)[1].

Figure 6.1 shows a comparison between RFM-SLAM and GTSAM [36] for a map with $\approx$ 2000 nodes. When the simulated data is input to GTSAM, the odometery based initial guess results in catastrophic failure as GTSAM gets stuck in a local minima whereas RFM-SLAM recovers the robot trajectory and map given identical data. The major contributions of this method can be summed up as follows:

[1] © 2017 IEEE. Reprinted, with permission, from Saurav Agarwal, Vikram Shree, and Suman Chakravorty. RFM-SLAM: Exploiting relative feature measurements to separate orientation and position estimation in slam. In Proc. IEEE International Conference on Robotics and Automation (ICRA)., May 29 - June 3 2017



(a) RFM-SLAM estimate for robot trajectory in one of our simulations. RMS position error is 1.44m.

(b) GTSAM estimate for robot trajectory for the same run, catastrophic failure due to bad initial guess.

Figure 6.1: Simulation results for map M2 with ≈ 2000 nodes for RFM-SLAM and GTSAM given identical data. The true trajectory is in green, **odometery** is in black, RFM-SLAM estimates are shown in blue and GTSAM estimates in magenta. Feature plots are omitted for the sake of clarity.

1. RFM-SLAM extends the class of methods that exploit two-step orientation and position estimation to feature-based SLAM.

2. Our approach reduces computational complexity of the optimization problem, i.e., if there are N poses where each pose $\mathbf{x}_k = [\mathbf{p}_k, \boldsymbol{\theta}_k]^T$ and L landmarks then we solve for N variables as opposed to $3N + 2L$ in existing methods (for the planar SLAM problem).

3. We show through empirical results that as odometery noise increases, our method's accuracy degrades gracefully compared to a state-of-the-art nonlinear optimization-based SLAM solver.

In Section 6.1 we state our problem and preliminaries, subsequently in Section 6.2 we present our approach wherein some mathematical details are relegated to Appendix 10.1. Results are discussed in Section 6.3 followed by conclusions.

6.1 Preliminaries and Problem

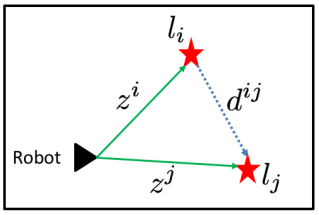
A portion of preliminaries presented in this section overlap with previous work in Chapter 5 but are provided here for the sake of completeness and clarity. We define ${}^{l}\mathbf{d}_k^{ij}$ to be the relative feature measurement, from feature l_i to l_j in the local frame of the robot at time t_k. The local relative measurement is computed as ${}^{l}\mathbf{d}_k^{ij} = {}^{l}\boldsymbol{\Delta}_k^{j} - {}^{l}\boldsymbol{\Delta}_k^{i}$, where ${}^{l}\boldsymbol{\Delta}_k^{i}, {}^{l}\boldsymbol{\Delta}_k^{j}$ are relative positions of features l_i and l_j respectively with respect to the robot in its local frame. A local measurement in the robot frame can be projected into the world (global) frame as

$$\mathbf{C}(\boldsymbol{\theta}_k)^T \, {}^{l}\boldsymbol{\Delta}_k^{i} = {}^{w}\boldsymbol{\Delta}_k^{i} = \mathbf{l}_i - \mathbf{p}_k, \tag{6.1}$$

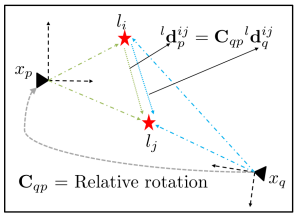
where $\mathbf{l}_i$ and $\mathbf{p}_k$ are the feature and robot positions in the world frame. Thus, it is the transformation of local measurements to the global frame that introduces non-linearity due to the trigonometric functions of orientation. If heading $\boldsymbol{\theta}^*$ is known, define ${}^{l}\boldsymbol{\Delta}$ to be the vector of all local feature position measurements and let $[\mathbf{p}^T \; \mathbf{l}^T]^T$ be the vector of all robot and feature positions in the world frame, then we have the following standard linear estimation problem in position

$$\mathbf{C}(\boldsymbol{\theta}^*)^T \, {}^{l}\boldsymbol{\Delta} = \mathbf{A}' \begin{bmatrix} \mathbf{p} \\ \mathbf{l} \end{bmatrix}, \tag{6.2}$$

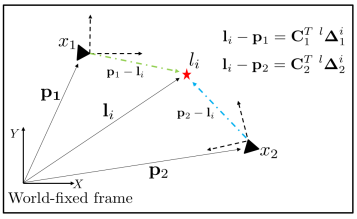
where $\mathbf{A}'$ is a matrix composed of elements in the set $\{-1, 0, 1\}$. However, direct heading estimates may not be readily available due to which we need to estimate the robot heading. In the proceeding section we develop the RFM-SLAM algorithm and describe our heading and position estimation method in detail. It is assumed that relative orientation measurements are independent, all noises are assumed to be zero-mean Gaussian and the front-end is given, the focus of this chapter is on the



(a) Robot making local relative measurements.



(b) Robot observes same features from two different poses forming a relative rotation constraint.



(c) Tranformation of local robot to feature relative measurements to the global frame.

Figure 6.2: (a) A robot making observations to two features l_i and l_j at time t_k, the range bearing measurements allow the robot to compute the relative positions ${}^l\Delta_k^i$ and ${}^l\Delta_k^j$ of the features in its local frame which are then transformed to a relative displacement measurement ${}^l d_k^{ij}$ between the two features. (b) A robot making observations to two features from poses x_p (green arrows) and x_q (blue arrows). Seeing the same two features forms a rotation constraint C_{qp} between these poses. (c) A robot sees the same landmark from two poses, the transformation of local relative measurements to the global frame is used in Section 6.2.3 to solve for robot and feature positions.

SLAM back-end estimation problem.

6.2 Methodology

The key steps in RFM-SLAM are as follows:

1. Transform range bearing observations from robot to features into relative position measurements in the robot's local frame at each pose, then calculate feature-to-feature displacements vectors (Section 6.2.1).

2. Compute the relative rotation constraints for poses that either are connected by proprioceptive odometery or view identical pairs of landmarks or both (Section 6.2.2.1).

3. Compute the Maximum Likelihood Estimate (MLE) for the robot orientation given constraints computed in the previous step (Section 6.2.2.2).

4. Solve the global linear estimation problem over robot and feature positions (Section 6.2.3).

6.2.1 Relative Feature Displacement Estimation

Figure 6.2 depicts our proposed feature mapping process. As the robot moves, it makes range bearing measurements at each pose to visible features. These range bearing measurements allow the robot to compute relative positions of landmarks in its local frame. These relative positions are converted into relative displacement measurements between features. At time t_k let the robot make range bearing measurements z_k^i and z_k^j to landmarks l_i and l_j respectively. Using the inverse measurement model g (Section 5.1), we can compute the displacement vector from l_i to l_j in the robot frame as given by Eq. 5.7.

${}^l\mathbf{g}_d(\mathbf{z}_k^j, \mathbf{z}_k^i)$ is the relative measurement from l_i to l_j in the robot's frame, which is independent of robot position and orientation. Figure 6.2(a) shows a simple depiction of a robot making a relative position measurement between two features. Let ${}^l\hat{\mathbf{d}}_k = {}^l\mathbf{g}_d(\mathbf{z}_k)$ be the vector of local relative measurements, $\bar{\nabla}{}^l\mathbf{g}_d|_{\mathbf{z}_k}$ be the Jacobian of function ${}^l\mathbf{g}_d(\mathbf{z}_k)$ and $\mathbf{R}_{\mathbf{z}_k}$ be the noise covariance of $\mathbf{z}_k$. Then we have ${}^l\hat{\mathbf{d}}_k \sim \mathcal{N}({}^l\mathbf{d}_k, {}^l\mathbf{R}_{\mathbf{d}_k} = \bar{\nabla}{}^l\mathbf{g}_d|_{\mathbf{z}_k}\mathbf{R}_{\mathbf{z}_k}\bar{\nabla}{}^l\mathbf{g}_d^T|_{\mathbf{z}_k})$. It is important to note that though measurements to each feature are independent, the set of relative feature measurements is *correlated*. This can be attributed to the correlations between relative measurements from common landmarks.

6.2.2 Heading Estimation

We now proceed to develop a two-part heading estimation technique. First, we recognize the fact that relative feature measurements-based constraints on the rotation between two poses are linear in the elements of relative orientation Direction Cosine Matrix (DCM). Thus we propose a linear least squares formulation to estimate

the relative rotation between poses. The second step is described in Section 6.2.2.2 where we apply an on-manifold optimization approach to solve the general non-linear heading estimation problem at loop closure given relative orientation estimates.

6.2.2.1 Linear Relative Rotation Estimation

Let $\mathbf{C}_{qp} = \mathbf{C}_p \mathbf{C}_q^T$ be the relative rotation matrix between two poses x_p, x_q where $\mathbf{C}_p, \mathbf{C}_q \in \mathrm{SO}(2)$. Let there be two landmarks l_i, l_j visible from poses x_p and x_q such that ${}^l\mathbf{d}_p^{ij}$ and ${}^l\mathbf{d}_q^{ij}$ are respectively the relative feature measurements in the local frame at each pose. Thus we have a constraint ${}^l\mathbf{d}_p^{ij} - \mathbf{C}_{qp}{}^l\mathbf{d}_q^{ij} = \mathbf{0}$ for every such pair of landmarks. Let $\mathbf{c}_{qp} \in \mathbb{R}^2$ be the vector of parameters for $\mathbf{C}_{qp}$ (see Eq. 10.1, Appendix 10.1).

As a robot moves, it makes two types of noisy observations:

1. Proprioceptive odometery measurements $\widehat{\delta\theta}_{odo} \sim \mathcal{N}(\delta\theta, \sigma_{odo}^2)$ provide a direct estimate of the relative rotation angle $\delta\theta$ between successive poses x_p and $x_{q=p+1}$, where σ_{odo}^2 is the measurement noise variance.

2. Relative feature measurements to common landmarks from two poses provide a relative orientation constraint. Let ${}^l\hat{\mathbf{d}}'_p, {}^l\hat{\mathbf{d}}'_q$ be the respective local relative measurements to common features from made from poses x_p and x_q, with error covariances ${}^l\mathbf{R}'_{\mathbf{d}_p}, {}^l\mathbf{R}'_{\mathbf{d}_q}$ respectively. We have the following linear problem for the relative rotation parameter vector $\mathbf{c}_{qp}$,

$$ {}^l\hat{\mathbf{d}}'_p = \mathbf{B}'_{qp}\mathbf{c}_{qp} + \mathbf{v}_{\mathbf{d}_{pq}}, \tag{6.3} $$

where $\mathbf{B}'_{qp}({}^l\hat{\mathbf{d}}'_q)$ (see Eq. 10.2, Appendix 10.1) is a matrix function of the relative measurements from pose x_q and $\mathbf{v}_{\mathbf{d}_{pq}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{\mathbf{d}_{pq}})$ is a zero-mean Gaussian measurement noise. The error covariance in this measurement is approximated

as $\mathbf{R}_{\mathbf{d}_{pq}} = {}^{l}\mathbf{R}'_{\mathbf{d}_p} + \hat{\mathbf{C}}_{qp,init}\,{}^{l}\mathbf{R}'_{\mathbf{d}_q}\hat{\mathbf{C}}^T_{qp,init}$.

For successive poses, $\hat{\mathbf{C}}_{qp,init} = \hat{\mathbf{C}}_{qp,odo}$, i.e., the relative rotation estimate from proprioceptive odometery. Between successive poses, all feature constraints in the form of Eq. 6.3 can be stacked along with proprioceptive odometery measurements which gives us the following linear problem

$$\begin{aligned}\begin{bmatrix}\hat{\mathbf{c}}_{qp,odo}\\ {}^{l}\hat{\mathbf{d}}_p\end{bmatrix} &= \begin{bmatrix}\mathbf{I}\\ \mathbf{B}'_{qp}\end{bmatrix}\mathbf{c} + \begin{bmatrix}\mathbf{v}_{\mathbf{c}_{qp,odo}}\\ \mathbf{v}_{\mathbf{d}_{pq}}\end{bmatrix}\\ &= \mathbf{B}_{qp}\mathbf{c} + \mathbf{v}_{\mathbf{c}_{pq}}\end{aligned} \tag{6.4}$$

where $\mathbf{v}_{\mathbf{c}_{pq}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{\mathbf{c}_{qp}})$, and $\mathbf{R}_{\mathbf{c}_{qp}} = diag([\mathbf{R}_{\mathbf{c}_{qp,odo}}, \mathbf{R}_{\mathbf{d}_{pq}}])$. Equation 6.4 can be rewritten as

$$\mathbf{z}_{\mathbf{c}_{qp}} = \mathbf{B}_{qp}\mathbf{c}_{qp} + \mathbf{v}_{\mathbf{c}_{qp}}. \tag{6.5}$$

Dropping the pose subscript for clarity, we can compute the estimate

$$\hat{\mathbf{c}} = (\mathbf{B}^T\mathbf{R}_{\mathbf{c}}^{-1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{R}_{\mathbf{c}}^{-1}\mathbf{z}_{\mathbf{c}} \tag{6.6}$$

and its error covariance

$$\mathbf{\Sigma}_{\mathbf{c}} = (\mathbf{B}^T\mathbf{R}_{\mathbf{c}}^{-1}\mathbf{B})^{-1}. \tag{6.7}$$

A robot may close a loop and return to a previously visited location and re-observe features. At loop closure, we may solve Eq. 6.3 to estimate the relative rotation between two poses x_p and x_q. In this case, $\hat{\mathbf{C}}_{qp,init} = \hat{\mathbf{C}}_p\hat{\mathbf{C}}_q^T$, where $\hat{\mathbf{C}}_p, \hat{\mathbf{C}}_q^T$ are

estimated by chaining together successive relative rotation estimates. To ensure that the solution is an orthogonal rotation, we project it back onto the SO(2) manifold as $\hat{\mathbf{c}}_{proj} = \boldsymbol{\eta}(\hat{\mathbf{c}})$, where $\boldsymbol{\eta}$ is a vector valued function (see Eq. 10.4, Appendix 10.1). The error covariance post projection is $\boldsymbol{\Sigma}_{\mathbf{c}_{proj}} = \bar{\nabla}\boldsymbol{\eta}|_{\hat{\mathbf{c}}}\Sigma_{\mathbf{c}}\bar{\nabla}^T\boldsymbol{\eta}|_{\hat{\mathbf{c}}}$ where $\bar{\nabla}\boldsymbol{\eta}|_{\hat{\mathbf{c}}}$ is the Jacobian of projection function $\boldsymbol{\eta}$ computed at the estimated values. Once $\hat{\mathbf{c}}$ is computed, it is transformed into the relative heading angle value (Eq. 10.5, Appendix Appendix 10.1), which in 2D is the scalar $\widehat{\delta\theta}$.

Planar SLAM has the property that relative orientation measurements are linear in heading by virtue of which we can formulate the following linear problem

$$\widehat{\boldsymbol{\delta\theta}} = \mathbf{H}\boldsymbol{\theta} + \mathbf{v}_\theta, \tag{6.8}$$

where $\widehat{\boldsymbol{\delta\theta}}$ is the vector all relative orientation measurements, $\mathbf{H}$ is a matrix composed of elements from the set $\{-1, 0, +1\}$ and $\boldsymbol{\theta}$ is the vector of robot heading angles. However, solving Eq. 6.8 directly may not provide the correct answer as the linear least squares formulation is indifferent to the angle wrap-around problem. In the proceeding section we describe how to overcome this problem. Lastly, we may compute the information matrix of the global heading estimate from Eq. 6.8 as $\boldsymbol{\Omega}_{\boldsymbol{\theta}} = \mathbf{H}^T\mathbf{R}_\theta^{-1}\mathbf{H}$ where $\mathbf{R}_\theta$ is a diagonal matrix composed of uncertainty in relative orientation estimates. In Section 6.2.3 we show how information matrix $\boldsymbol{\Omega}_{\boldsymbol{\theta}}$ is used by our algorithm to compute the map and history of robot positions.

6.2.2.2 On-Manifold Optimization Using Relative Orientation Measurements

The method described previously allows us to estimate relative rotations between poses. The set of poses and constraints from relative rotation estimates form a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose nodes $\mathcal{V} = \{\nu_1, \ldots, \nu_n\}$ are the pose orientations and whose edge $\epsilon_{pq} \in \mathcal{E}$ is a relative orientation constraint between nodes ν_p, ν_q. The problem

at hand is to compute the global orientations for all nodes given relative rotation measurements.

Let $\hat{\mathbf{C}}_{qp}$ be the estimate of DCM $\mathbf{C}_{qp}$ for the relative rotation between nodes ν_p, ν_q. In the noise free measurement case, $\hat{\mathbf{C}}_{qp}\mathbf{C}_q = \mathbf{C}_p$. However, given a set of noisy measurements we minimize $\sum_{\epsilon_{pq}\in\mathcal{E}} \kappa_{qp}||\hat{\mathbf{C}}_{qp}\mathbf{C}_q - \mathbf{C}_p||_F$ where $||\cdot||_F$ denotes the Frobenius matrix norm and κ_{qp} is a weight for the measurement $\hat{\mathbf{C}}_{qp}$. Now the Frobenius norm can be expanded as $||\hat{\mathbf{C}}_{qp}\mathbf{C}_q - \mathbf{C}_p||^2 = ||\mathbf{C}_p||^2 + ||\mathbf{C}_q||^2 - 2\text{tr}(\mathbf{C}_q^T\hat{\mathbf{C}}_{qp}^T\mathbf{C}_p)$.

Thus minimizing the Frobenius norm is equivalent to minimizing the term $-\text{tr}(\mathbf{C}_q^T\hat{\mathbf{C}}_{qp}^T\mathbf{C}_p)$ where $\text{tr}(\cdot)$ denotes the trace operator. Using properties of trace ($\text{tr}(\mathbf{X}) = \text{tr}(\mathbf{X}^T)$), we have the cost function to minimize as

$$J = -\sum_{\epsilon_{pq}\in\mathcal{E}} \kappa_{qp}\ \text{tr}(\mathbf{C}_p^T\hat{\mathbf{C}}_{qp}\mathbf{C}_q), \tag{6.9}$$

where $\kappa_{qp} = 1/\sigma_{\delta\theta_{qp}}$, i.e., inverse of standard deviation of relative rotation estimate. The Euclidean gradients for the cost function J are

$$\frac{\partial J}{\partial \mathbf{C}_p} = -\sum_{\epsilon_{pq}\in\mathcal{E}} \kappa_{qp}\hat{\mathbf{C}}_{qp}\mathbf{C}_q,\ \frac{\partial \mathcal{F}_c}{\partial \mathbf{C}_q} = -\sum_{\epsilon_{pq}\in\mathcal{E}} \kappa_{qp}\hat{\mathbf{C}}_{qp}^T\mathbf{C}_p. \tag{6.10}$$

Note that in the cost function given by Eq. 6.9, we directly optimize over the set of orientations for all poses. The initial guess can be computed by chaining together relative rotation estimates computed in Section 6.2.2.1. Another way of looking at Eq. 6.9 is as follows, we have $\hat{\mathbf{C}}_{qp} = \mathbf{V}_{qp}\mathbf{C}_p\mathbf{C}_q$ where $\mathbf{V}_{qp}$ is the perturbation due to noise. Then solving Eq. 6.9 is equivalent to computing the maximum likelihood estimator with a Langevin prior on the perturbation $\mathbf{V}_{qp}$ where κ_{qp} becomes the Langevin concentration parameter [18]. We use the Manopt MATLAB toolbox developed in [19] to minimize the cost function J using trust regions based optimization

routine [7].

6.2.3 Global Trajectory and Feature Estimation

Abusing notation slightly, let ${}^{l}\hat{\mathbf{\Delta}} \sim \mathcal{N}({}^{l}\mathbf{\Delta}, {}^{l}\mathbf{R}_{\mathbf{\Delta}} = blkdiag([{}^{l}\mathbf{R}_{\mathbf{\Delta}_1}, {}^{l}\mathbf{R}_{\mathbf{\Delta}_2} \dots]))$ be the vector of all local relative position measurements, i.e. from robot to features and odometery between successive poses. After computing the global orientations according to Section 6.2.2.2, the vector of local relative measurements ${}^{l}\hat{\mathbf{\Delta}}$ can be transformed to the world frame similar to Eq. 6.2. From the transformed global measurements we can formulate the linear estimation problem as

$$ {}^{w}\hat{\mathbf{\Delta}} = \hat{\mathbf{C}}^{T}\ {}^{l}\hat{\mathbf{\Delta}} = \mathbf{A}' \begin{bmatrix} \mathbf{p} \\ \mathbf{l} \end{bmatrix} + {}^{w}\mathbf{v}_{\mathbf{\Delta}}, \tag{6.11} $$

where $\hat{\mathbf{C}} = \mathbf{C}(\hat{\boldsymbol{\theta}})$ is the corresponding composition of DCM matrices parametrized by the estimated heading $\hat{\boldsymbol{\theta}}$, $[\mathbf{p}^T\ \mathbf{l}^T]^T$ is the vector of robot and feature positions, $\mathbf{A}'$ is a matrix with each row containing elements of the set $\{-1, 0, +1\}$ and ${}^{w}\mathbf{v}_{\mathbf{\Delta}} \sim \mathcal{N}(\mathbf{0}, {}^{w}\mathbf{R}_{\mathbf{\Delta}} = \mathbf{C}^{T\,l}\mathbf{R}_{\mathbf{\Delta}}\mathbf{C})$ is the noise vector. If we were to solve for positions directly from Eq. 6.11, we would end up with an incorrect estimate as the global orientation estimates $\hat{\boldsymbol{\theta}}$ are correlated. Thus relative position measurements in the global frame are correlated with heading estimates as well. We now describe how to setup the position estimation problem while correctly incorporating the appropriate error covariances similar to the trick employed in LAGO [24]. After computing the orientation estimates $\hat{\boldsymbol{\theta}}$ along with the transformed global relative position measurements we stack them to give us a new measurement vector $\boldsymbol{\gamma}$. Then we have

$$\boldsymbol{\gamma} = \mathbf{h}_w({}^l\boldsymbol{\Delta}, \boldsymbol{\theta}) + \mathbf{v}_w = \begin{bmatrix} \hat{\mathbf{C}}^T \, {}^l\hat{\boldsymbol{\Delta}} \\ \hat{\boldsymbol{\theta}} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \mathbf{p} \\ \mathbf{l} \\ \boldsymbol{\theta} \end{bmatrix} + \begin{bmatrix} {}^w\mathbf{v}_{\boldsymbol{\Delta}} \\ \mathbf{v}_{\boldsymbol{\theta}} \end{bmatrix}. \tag{6.12}$$

The error covariance $\mathbf{R}_{\boldsymbol{\gamma}}$ of measurement vector $\boldsymbol{\gamma}$ is then given by,

$$\mathbf{R}_{\boldsymbol{\gamma}} = \bar{\nabla}\mathbf{h}_w \begin{bmatrix} {}^l\mathbf{R}_{\boldsymbol{\Delta}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \bar{\nabla}^T\mathbf{h}_w \tag{6.13}$$

where $\bar{\nabla}\mathbf{h}_w$ is the Jacobian of measurement function $\mathbf{h}_w$ (Eq. 6.12) given by

$$\bar{\nabla}\mathbf{h}_w = \begin{bmatrix} \mathbf{C}^T & \mathbf{M} \, {}^l\hat{\boldsymbol{\Delta}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{6.14}$$

where $\mathbf{M} = \frac{\partial \mathbf{C}^T}{\partial \boldsymbol{\theta}}$. Thus we have

$$\mathbf{R}_{\boldsymbol{\gamma}} = \begin{bmatrix} {}^w\mathbf{R}_{\boldsymbol{\Delta}} + \mathbf{M}\boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{M}^T & \mathbf{M}\boldsymbol{\Sigma}_{\boldsymbol{\theta}} \\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{M}^T & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix}. \tag{6.15}$$

Finally, the solution to the linear estimation problem of Eq. 6.12 is given by

$$\begin{bmatrix} \mathbf{p}^* \\ \mathbf{l}^* \\ \boldsymbol{\theta}^* \end{bmatrix} = (\mathbf{A}^T\mathbf{R}_{\boldsymbol{\gamma}}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{R}_{\boldsymbol{\gamma}}^{-1}\boldsymbol{\gamma}. \tag{6.16}$$

Note that Eq. 6.16 involves the inversion of a large sparse matrix $\mathbf{R}_{\boldsymbol{\gamma}}$ which may not be suitable for implementation due to complexity and potential numerical issues. However, this inversion is easily avoided by analytically computing the information matrix $\boldsymbol{\Omega}_{\boldsymbol{\gamma}} = \mathbf{R}_{\boldsymbol{\gamma}}^{-1}$ using block-matrix inversion rules as

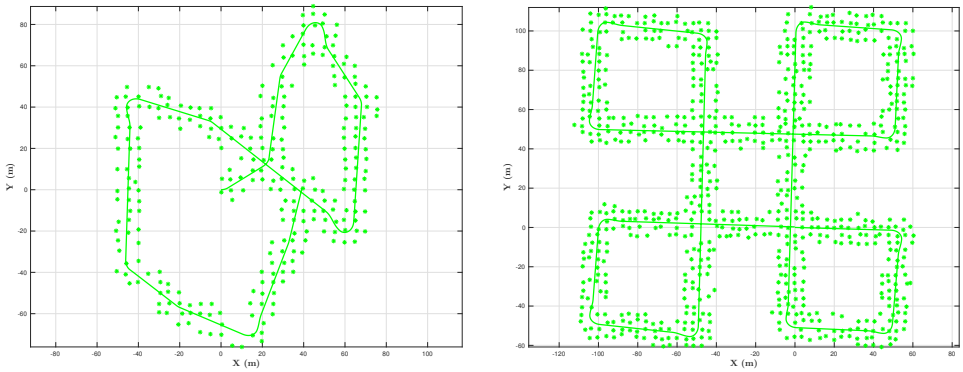
$$\boldsymbol{\Omega}_{\boldsymbol{\gamma}} = \begin{bmatrix} {}^{w}\mathbf{R}_{\boldsymbol{\Delta}}^{-1} & -{}^{w}\mathbf{R}_{\boldsymbol{\Delta}}^{-1}\mathbf{M} \\ -\mathbf{M}^{T\,w}\mathbf{R}_{\boldsymbol{\Delta}}^{-1} & \boldsymbol{\Omega}_{\boldsymbol{\theta}} + \mathbf{M}^{T\,w}\mathbf{R}_{\boldsymbol{\Delta}}^{-1}\mathbf{M} \end{bmatrix}. \tag{6.17}$$

6.3 Results

We conducted 1600 simulations in total for two planar maps M1 and M2 (see Figs. 6.3(a) and 6.3(b)). The maps themselves were constructed by randomly sampling landmarks in a 2D environment after which simulated sensor data was collected by driving the robot around a sequence of waypoints. Proprioceptive odometery noise σ_{odo} is varied by scale factor $\alpha = \{1, 2, 3, 4\}$, where $\alpha = 1$ corresponds to $\sigma_{odo} = diag([0.05\text{m}, 0.05\text{m}, 0.6^{\circ}])$ and range bearing sensor noise σ_{rb} is varied by scale factor $\beta = \{1, 2, 3, 4\}$, where $\beta = 1$ corresponds to $\sigma_{rb} = diag([0.05\text{m}, 0.6^{\circ}])$. For each map, 50 simulations were conducted for each fixed noise level and 16 variations of noise values were used in total. For each simulation the resulting data was processed by both RFM-SLAM (MATLAB) and GTSAM (C++) [36]. GTSAM utilized the Levenberg-Marquardt Algorithm and both Manopt [19] and GTSAM were allowed a maximum of 100 iterations. We now proceed to discuss our results in the context of key aspects that affect solution accuracy, i.e., map, odometery noise and range bearing sensor accuracy.

6.3.1 Changing Map

Figure 6.3 shows the two maps; map M1 with 1129 robot poses and 286 landmarks; and map M2 with 2064 robot poses and 777 features. Each map presents a different challenge, i.e., in M1 there are 2 loop closures and robot trajectory does not terminate at the start location, whereas in M2 there are 5 loop closures and robot returns to its start location. Table 6.1 shows that GTSAM average RMSE in robot pose is greater for map M2 than M1 for all noise combinations except for

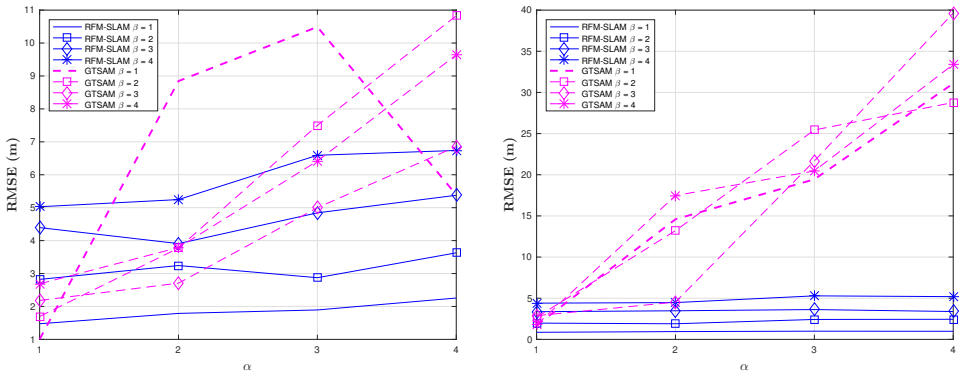


(a) Map M1 with 1129 robot poses and 286 landmarks. The robot trajectory is 544.50m long with 2 loop closures but robot does not return to start.

(b) Map M2 with 2064 robot poses and 777 landmarks. The robot trajectory is 1000.87m long with 5 loop closures.

Figure 6.3: The two scenarios used in the simulations and comparisons.

$\alpha = 1, \beta = 4$. We note GTSAM suffers more catastrophic failures in map M2 than map M1 (Table 6.1). This is despite the fact that there are more loop closures in M2 and robot returns to start. The previous observation may be attributed to the trajectory in M2 ($\approx$ 1000m) being longer than in M1 ($\approx$ 500m) which results in odometery based initial guess being further from the ground truth than for map M1. An interesting difference emerges, for all noise combinations in the case of RFM-SLAM, average RMSE for map M2 is smaller than that for M1 despite the trajectory in M2 being twice as long as that of M1. This may be attributed to two factors; (i) RFM-SLAM is able to exploit the graph topology for M2 [78] (multiple loop closures) in the orientation estimation phase; (ii) using range bearing measurements to augment relative orientation estimation provides a measure of robustness to the on-manifold optimization problem and purely odometery-based initial guess plays no role in the estimation process.



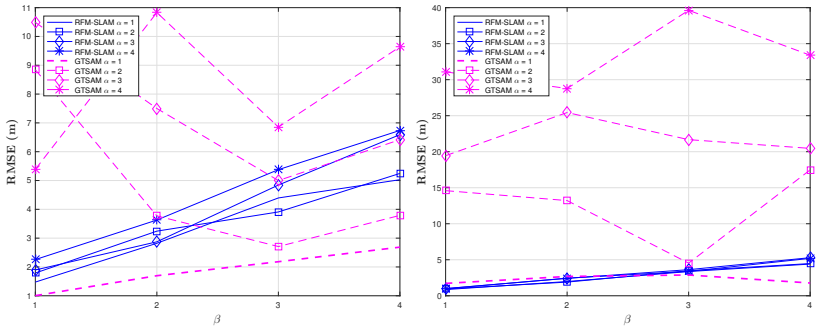
(a) RMSE vs. α for map M1.

(b) RMSE vs. α for map M2.

Figure 6.4: Behavior of RMSE in robot position as odometery noise level α is increased for different β. The solid blue curves depict RFM-SLAM behavior and dashed magenta curves are for GTSAM.

6.3.2 Increasing Proprioceptive Odometery Noise

Figure 6.4 shows that for low odometery noise $\alpha = 1$ both methods perform comparably (same order of magnitude in RMSE) in both scenarios. For $\alpha = 1$, in the case of map M1 GTSAM performs better than RFM-SLAM for low range bearing noise. Increasing the proprioceptive odometery noise has the effect of reducing the quality of initial guess that GTSAM relies on which is evident from Figs. 6.4(a) and 6.4(b). In both maps, as α is increased, RFM-SLAM performance degrades much slower compared to GTSAM, where in map M2 particularly (Fig. 6.5(b)) GTSAM shows a rapid decline in solution accuracy. We take the case of $\beta = 1$ to highlight the variation in solution accuracy as odometeric noise is increased from lowest ($\alpha = 1$) to its highest ($\alpha = 4$) value. In the case of M1, GTSAM solution accuracy degrades from 1.004m to 5.389m as the number of catastrophic failures increased from 0 to 4, whereas RFM-SLAM accuracy degrades from 1.475m to 2.256m. In the case of M2, GTSAM solution accuracy degrades rapidly by 1709.3% as RMSE rises from 1.718m to 31.084m due to the number of catastrophic failures rising from 1 to 18 whereas



(a) RMSE vs. β for map M1. (b) RMSE vs. β for map M2.

Figure 6.5: Behavior of RMSE in robot position as range bearing noise level β is increased for different α.

RFM-SLAM accuracy reduces gently from 0.859m to 0.982m. Thus simulation results show that RFM-SLAM solution accuracy degrades gracefully for both maps with increasing noise as it does not suffer catastrophic failure whereas GTSAM's performance is dominated by its sensitivity to the initial guess error (odometery).

6.3.3 Increasing Range Bearing Sensor Noise

Figure 6.5 shows that for the lowest odometeric noise value ($\alpha = 1$), both methods show a well defined behavior in RMSE growth as β increases. We look at the variation in error between lowest ($\beta = 1$) and highest ($\beta = 4$) range bearing sensor noise when proprioceptive odometery noise is lowest ($\alpha = 1$). In map M1 as β increases from 1 to 4, RFM-SLAM RMSE rises from 1.475m to 5.028m, for GTSAM in the same map, we see a rise from 1.004m to 2.687m. In case of map M2, RFM-SLAM RMSE increases from 0.859m to 4.4m (418% increase) whereas for GTSAM we see a rise from 1.718m to 1.771m. Thus RFM-SLAM exhibits a higher relative increase in RMSE than GTSAM for increasing β. Thus simulation results show that compared to GTSAM, RFM-SLAM performance is dominated by range bearing sensor noise.

6.3.4 Discussion on Results

Each method has a dominating factor that affects its behavior; for RFM-SLAM it is the range bearing sensor noise as we rely on this information in the orientation optimization phase; for GTSAM it is the proprioceptive odometery as it relies on odometery to bootstrap the solver. However, our results indicate that for all noise values, RFM-SLAM remains free of catastrophic failures due to which RMSE growth behaves well unlike in the case of GTSAM where the propensity of catastrophic failures increases with odometeric noise. In the case of GTSAM we see an order of magnitude increase in maximum RMSE over RFM-SLAM ($\approx$ 40m vs. $\approx$ 7m) at $\alpha = 4, \beta = 3$. In few cases, GTSAM failed to converge to a solution, these numbers are also reported in Table 6.1. Further as the number of robot poses grows, odometery based initial guess diverges in an unbounded manner which may tend to dominate the solution accuracy in existing methods compared to noise in range bearing sensing.

6.4 Discussion

In Section 6.2.2.1, two approximations are involved. The first in calculation of error covariance of relative orientation constraints and the second in transformation from DCM parameters to a scalar angle value. Both involve a linearization that works well in practice as demonstrated by results in Section 6.3. However in cases where the initial guess is of very high-quality, standard non-linear optimization may provide better accuracy as this approximation is not involved (see Fig. 6.4).

Extending to 3D: The global orientation optimization problem given relative measurements (Eq. 6.9) does not change from 2D to 3D. A minor difference arises in solving for relative orientation at loop closure (Eq. 6.3) where a robot would require observations to 3 features from two poses as 9 constraints are required to solve for the

Scenario	Average RMSE in robot position estimate **(number of catastrophic failures, number of cases with no convergence)**							
	$\beta = 1$		$\beta = 2$		$\beta = 3$		$\beta = 4$	
	RFM-SLAM	GTSAM	RFM-SLAM	GTSAM	RFM-SLAM	GTSAM	RFM-SLAM	GTSAM
	$\alpha = 1$							
M1	1.475 **(0)**	1.004 **(0)**	2.822 **(0)**	1.696 **(0)**	4.395 **(0)**	2.182 **(0)**	5.028 **(0)**	2.687 **(0)**
M2	0.859 **(0)**	1.718 **(1)**	1.976 **(0)**	2.682 **(2)**	3.365 **(0)**	2.882 **(1)**	4.400 **(0)**	1.771 **(0)**
	$\alpha = 2$							
M1	1.789 **(0)**	8.846 **(9)**	3.243 (0)	3.772 **(2)**	3.909 **(0)**	2.708 **(0)**	5.246 **(0)**	3.790 **(0)**
M2	0.947 **(0)**	14.593 **(10)**	1.898 **(0)**	13.214 **(7)**	3.475 **(0)**	4.528 **(1)**	4.471 **(0)**	17.472 **(8,5)**
	$\alpha = 3$							
M1	1.894 **(0)**	10.489 **(10)**	2.873 **(0)**	7.500 **(7)**	4.843 **(0)**	4.999 **(3,1)**	6.592 **(0)**	6.417 **(3,3)**
M2	0.997 **(0)**	19.445 **(13)**	2.422 **(0)**	25.437 **(16,1)**	3.623 **(0)**	21.675 **(12,5)**	5.288 **(0)**	20.461 **(11,3)**
	$\alpha = 4$							
M1	2.256 **(0)**	5.389 **(4)**	3.629 **(0)**	10.830 **(7)**	5.377 **(0)**	6.856 **(3,1)**	6.738 **(0)**	9.640 **(7,5)**
M2	0.982 **(0)**	31.084 **(18,1)**	2.442 **(0)**	28.777 **(15,1)**	3.384 **(0)**	39.631 **(21,4)**	5.193 **(0)**	33.397 **(17,4)**

Table 6.1: Average robot position RMSE in meters with the number of catastrophic failures and number of instances where there was no solution in bold brackets as odometery and range bearing sensor noise are varied. For each noise and map combination, 50 simulations were conducted and the RMSE in pose estimation was averaged over these simulations (excluding runs for which GTSAM did not converge to a solution).

DCM ($\mathbf{C}_k \in \mathbb{R}^{3\times3}$ in SO(3)). Further, the linear position estimation problem of Eq. 6.12 also remains identical. The key difference occurs in computing the uncertainty over global orientation estimates as the 3D rotation problem cannot be setup similar to the 2D case (Eq. 6.8). In 3D, relative orientations measurements are not linear in robot orientation, rather they are non-linear functions of rotation parameters. In this regard, the work of [39] develops an analysis for first-order error propagation in 3D rotation estimation which may be applicable to future extension of this work.

A novel approach to solving the feature-based SLAM problem was presented that exploits separation of robot orientation from position estimation. Empirical results indicate that RFM-SLAM is able to avoid catastrophic failure and solution accuracy behaves well under varying noise conditions. Decoupling orientation estimation from position exhibits a distinct advantage in that robust solutions can be obtained which one may use to bootstrap full non-linear optimization solvers. Chapters 5 and 6 are limited in application to simulation based studies. Further a more common approach to SLAM is to use LiDAR in a scan-matching based approach for large scale SLAM. In the next chapter, develop an approach to 2D scan matching-based pose graph SLAM with absolute orientation sensing which is widely applicable to real-world settings.

7. POSE-GRAPH SLAM WITH ABSOLUTE ORIENTATION SENSING

It is well understood that a difficult problem in mobile robot autonomy is the challenge of long-term navigation using only on-board sensors in large GPS-denied environments. An important practical example are material handling robots that move goods (boxes, pallets etc.) in large warehouses and distribution centers. Since installing beacons, markers or guide cables is expensive, robots are often expected start without prior knowledge of the map. Therefore robust, accurate localization and mapping is necessary for reliable operation.

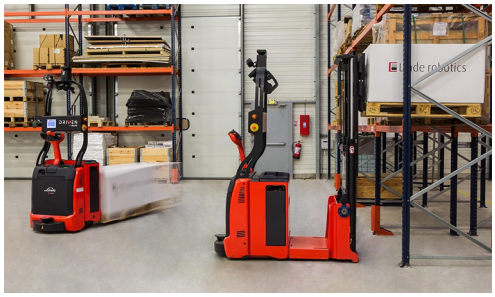


Figure 7.1: Robotic material handling in operation at a warehouse [1].

A common sensor for 2D mapping is LiDAR. In 2D LiDAR-based SLAM, robot designers overcome the unknown map problem by first driving a robot manually to gather and store measurements (e.g., laser scans) and then the process measurements offline to build detailed maps. A common technique to solve the SLAM problem is to use relative pose-graph SLAM. Pose-graph SLAM uses a two-pronged approach; (i) a front-end which maintains an estimate of the robot pose using correlative scan matching [108] or other suitable techniques and computes data association between current and past observations, and (ii) a back-end which solves the non-linear optimization

to compute the history of robot poses. Once the history of poses is computed, LiDAR scans are superimposed on the associated poses to build a map.

As discussed earlier, a major caveat of SLAM techniques is that to correct estimation drift they often rely on loop closure, i.e., revisiting previously seen locations and correctly associating sensor information to data previously stored in the map. Loop closure is *sensitive to data association accuracy* and wrong data associations can lead to catastrophic failure of the SLAM system as shown in Fig. 7.2(a) and previously in Chapter 6. Moreover, data association reliability is limited by localization accuracy. LiDAR sensors are known to be accurate in range, but scan matching (even when assisted by gyros) may often compute corrupt estimates of relative rotation between two poses. Thus localization drift causes map quality to degrade as the scale of environment increases, see Fig. 7.2(b).

Contributions: We propose a method for 2D SLAM that fuses absolute orientation sensing (using cameras that track stable structural features), with range-scan measurements using a LiDAR. Realistic simulation studies show that **our proposed method does not fail in mapping whereas existing state-of-the-art methods fail** $\approx 40\% - 50\%$ **of the times** (see Table 7.2). We conduct a detailed analysis of the effect of noisy relative orientation measurements and show that for long-term autonomy, absolute orientation measurements are critical to achieving robust localization and mapping. We demonstrate our approach successfully on a physical system in a real-world setting with a commercially available mobile robotics platform.

We call our approach LOGO-SLAM (Linear Orientation-based Graph Optimization for SLAM). Our proposed method is the first to demonstrate that a highly significant advantage, both in terms of localization accuracy, and consequently, robustness to failure, is achieved by directly using absolute orientation sensing in rela-

tive pose-graph SLAM. Furthermore, we claim that absolute orientation information may be robustly sensed in a variety of indoor scenarios by leveraging structural features. This concept is akin to star trackers that track far-off persistent features like star patterns to get global orientation in aerospace applications. For example, in industrial buildings, the ceiling lighting or corrugation is usually aligned along one direction. Thus, the true north of the building can be fixed in the direction of ceiling features and a vehicle can estimate its orientation with respect to the building by observing the ceiling. The benefit of this approach can be clearly seen in Fig. 7.2. For 3D applications, vertical columns or pillars may be used to infer roll and pitch information however we limit the scope of this work to 2D planar SLAM.

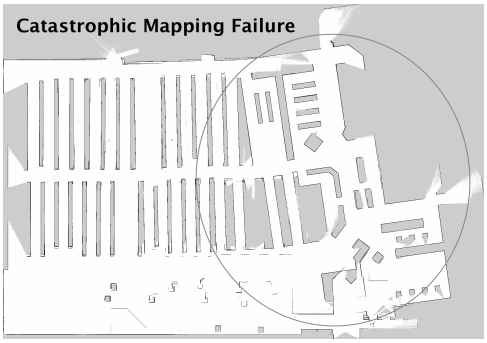
7.1 Preliminaries and Problem

Let $\mathbf{x} = \{\mathbf{x}_0, \ldots, \mathbf{x}_n\}$ be a set of $n+1$ poses, describing the robot position and orientation at each time k. In 2D (planar) problems, $\mathbf{x}_k = [\mathbf{p}_k^T \ \theta_k]^T \in \mathrm{SE}(2)$, where $\mathbf{p}_k \in \mathbb{R}^2$ is the position and $\theta_k \in \mathrm{SO}(2)$ is the heading. Let ξ_{ij} be a relative measurement of pose j w.r.t to pose i then,

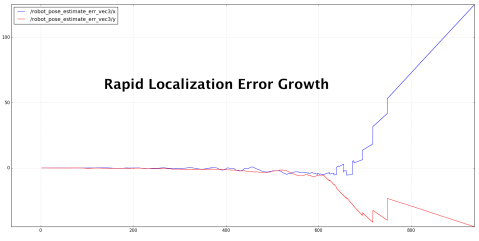
$$\xi_{ij} = \begin{bmatrix} {}^{l}\boldsymbol{\Delta}_{ij} = \mathbf{R}_i(\mathbf{p}_j - \mathbf{p}_i) \\ \delta\theta_{ij} = \theta_j - \theta_i \end{bmatrix} \tag{7.1}$$

where $\mathbf{R}_i$ is the rotation matrix composed by θ_i. In the general setting, ξ_{ij} is corrupted by noise, thus $\hat{\xi}_{ij} = \xi_{ij} + v_{ij}$, where v_{ij} is assumed to be zero-mean Gaussian. Let ${}^{l}\boldsymbol{\Delta}$ be the vector of relative position measurements in the local frame at each pose. If robot orientation θ^* is known at each pose, then the SLAM problem simply becomes

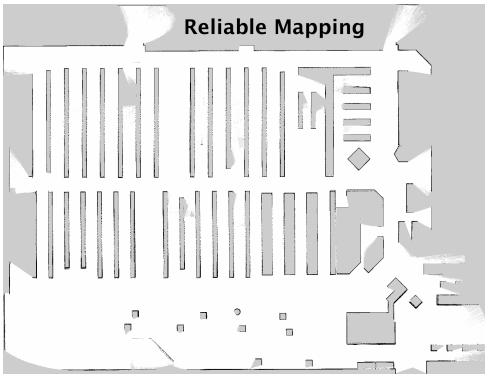
$${}^{w}\boldsymbol{\Delta} = \mathbf{R}(\boldsymbol{\theta}^*)^{\mathbf{T}} \ {}^{l}\boldsymbol{\Delta}. \tag{7.2}$$



(a) Catastrophic mapping failure results in a twisted map that is unusable by a robot for real-world tasks. Such failures are quite common with SLAM implementations.



(b) Localization error vs. time. Mapping failure is driven by localization error which usually results in erroneous data association in the SLAM front-end.



(c) Reliable mapping that is suitable to be used by a robot for real-world tasks.

(d) Localization error vs. time. Unlike in Fig. 7.2(b), the localization error is bounded.

Figure 7.2: Simulation result for existing state-of-the-art vs. proposed approach: Figures (a) and (b) show an example of mapping failure for a building with a floor area of 78,240 sq. ft. The floor plan is based on an HEB grocery store located in College Station, Texas. We used a state-of-the-art front-end [63], [108] and g2o [88] for the back-end. The solution failed in approximately 60% of the experiments that were run. Figures (b) and (c) show an example of successful mapping for the same environment. We used our novel approach which fuses absolute orientation information with a state of the art front-end [63], [108] and our back-end graph solver. Our approach succeeded in every one of the experiments that were run.

We know that ${}^{w}\boldsymbol{\Delta} = \mathbf{A}'\mathbf{p}$ where where $\mathbf{A}'$ is a matrix composed of elements in the set $\{-1, 0, 1\}$ and $\mathbf{p}$ is the vector of robot positions in the global frame. Thus, when robot orientation $\boldsymbol{\theta}$ is known, the position estimation problem is linear. Moreover, when unbiased global heading measurements are available, the problem can be very accurately linearized. This is the key insight that provides our method with high localization accuracy providing unparalleled robustness in the front-end and enabling *computationally low-cost linear-least squares solution* for the back-end. The same concept generalizes to 3D problems where the relative orientation measurement $\theta_j - \theta_i$ may be represented as $\mathbf{R}_j \mathbf{R}_i^T$.

7.2 Method

We now proceed to describe our approach in detail. Our method comprises three key aspects:

1. Using commonly occurring structural cues to sense absolute orientation of the robot. In an indoor setting, absolute orientation is measured relative to the building North.

2. Fusing absolute orientation measurements to the SLAM front-end, i.e., in our case to a scan matching algorithm.

3. Solving a batch optimization problem, to compute global estimates at loop closure by fusing relative pose measurements and absolute orientation sensing.

The next section deals with the issue of orientation sensing. In Section 7.2.2 we describe how the heading measurements are fused with the front-end in a filtering based scheme. Section 7.2.3 describes the batch optimization method to solve for loop closure and finally in Section 7.2.4 a discussion is presented for our approach.

7.2.1 Absolute Orientation Sensing

We seek independent absolute orientation estimates of the robot heading. In an indoor environment, we may use the relative heading of the robot w.r.t to the building's fixed North. Our orientation sensing method works by detecting structural features of the environment. In most indoor environments, e.g., offices, factories, warehouses etc. the ceiling structure usually has straight line features that are easy to detect. For example, ceiling corrugation in most industrial buildings is aligned along one direction which can be detected by a ceiling facing camera.

Several works in the field of vision-based SLAM [55, 132, 42, 151] have demonstrated the advantages of tracking line (edge) features. Compared to these methods, we do not explicitly map line features as part of the SLAM process. In [32] the authors present a method to detect camera orientation from a monocular image by assuming Manhattan world constraints on the environment. An in-depth discussion of these contributions is beyond the scope of this work. However we note that structural line features fall into two main categories, the horizontal group (ceiling corrugation, lighting, floor tiles etc.) and the vertical group (pillars, door frames, wall corners etc.). Lines in the horizontal group may be used to provide heading information and vertical lines are quite reliable for deducing roll and pitch. In this work, we restrict ourselves to tracking horizontal line features to determine heading. Algorithm 4 describes the process of estimating ceiling direction.

Figure 7.8(c) shows a polar plot of the histogram generated in Algorithm 4 in one of our simulation runs. Note that line direction is inherently ambiguous, i.e., it may not be possible to differentiate North from South (likewise from East to West). Therefore gyro data is used in the intermediate time between absolute orientation measurements to propagate the robot heading. Gyros usually provide data at >

Algorithm 4: Orientation Estimation from Structural Cues

Input: I, W, b
I is the input image
$l \leftarrow$ Extract line features from image I
$[\theta_1, \theta_2, \ldots, \theta_n] \leftarrow$ Compute orientation of the lines in local frame of image I
$h \leftarrow$ Create histogram of the orientation data with bins of width b in range $[0, 2\pi)$
$i_{max} \leftarrow$ Get index of bin with maximum measurements
$h_W \leftarrow$ Extract a window of width W around the bin i_{max}
$\theta_c, \sigma^2_{\theta_c} \leftarrow$ Compute the weighted mean of observations in window
$\theta_c \leftarrow$ Wrap θ_c in range $[0, \pi)$
return $\theta_c, \sigma^2_{\theta_c}$;

100Hz and therefore can be used to account for the angle wrap-around issue in absolute orientation detection. To estimate the robot heading, initial heading at time t_0 is assumed to be known (usually $\theta_0 = 0$). In the proceeding section, we describe how absolute orientation measurements are fused in the SLAM front-end.

7.2.2 Heading Assisted Front-End

We develop a modified version of the correlative scan matching method of [108] for our front-end. To the scan matching based front-end we add a Kalman filter after the scan match step. This Kalman filter fuses relative orientation estimates from scan matching with absolute orientation estimates as computed previously (using a ceiling facing camera and gyro). Let $\bar{\theta}_k \sim \mathcal{N}(\theta_k, \sigma^2_{\bar{\theta}_k})$ be the absolute orientation measurement at time t_k and $\hat{\theta}^+_{k-1}$ be the heading estimate of the robot at t_{k-1}. Let $\delta\hat{\theta}_{k-1,k} \sim \mathcal{N}(\delta\theta_{k-1,k}, \sigma^2_{\delta\theta_{k-1,k}})$ be the relative orientation measurement from pose x_{k-1} to x_k as computed by scan matching, Then $\hat{\theta}_k$ is computed as follows.

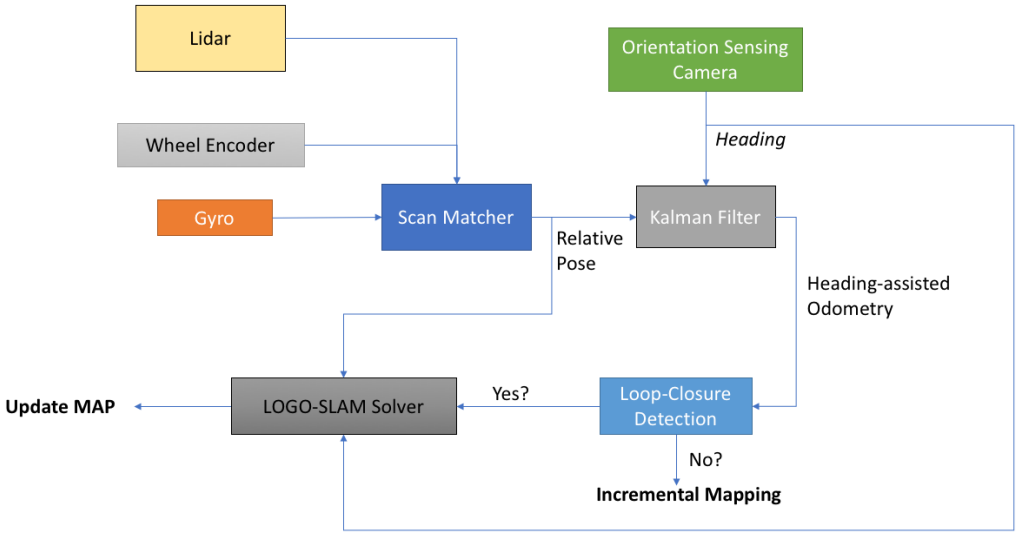


Figure 7.3: The LOGO-SLAM architecture. A flowchart depicting how sensor data flows and the various computation modules.

First the relative orientation estimate is used to compute a prediction,

$$\begin{aligned} \hat{\theta}_k^- &= \hat{\theta}_{k-1}^+ + \delta\hat{\theta}_{k-1,k} \\ {\sigma^2}_{\theta_k}^- &= {\sigma^2}_{\theta_{k-1}}^+ + \sigma^2_{\delta\theta_{k-1,k}} \end{aligned} \tag{7.3}$$

Then, the absolute orientation measurement is used to compute the update as follows,

$$\begin{aligned} \tilde{y}_k &= \bar{\theta}_k - \hat{\theta}_k^- \\ s_k &= \sigma^2_{\bar{\theta}_k} + {\sigma^2}_{\theta_k}^- \\ K_k &= {\sigma^2}_{\theta_k}^- s_k^{-1} \\ \hat{\theta}_k^+ &= \hat{\theta}_k^- + K_k\tilde{y}_k \\ {\sigma^2}_{\theta_k}^+ &= (1 - K_k){\sigma^2}_{\theta_k}^- \end{aligned} \tag{7.4}$$

At the update step, we use an innovation filter as a form of error checking. This is due to the fact that an erroneous scan match solution can produce corrupt yet confident (inconsistent) estimate of relative pose transformation. If passed into the back-end, an inconsistent relative pose estimate can result in egregious localization and mapping errors. Thus at time t_k if the innovation signal $\tilde{y}_k > \psi$, where ψ is a user defined threshold parameter, then the scan match is rejected and the robot relies purely on wheel odometry to update its localization estimate at time t_k.

7.2.3 Backend

Our SLAM back-end uses the graph generated by the front-end along with absolute orientation data and solves a two step optimization problem. The first step is the estimation of robot orientation using the absolute orientation and relative orientation measurements followed by a second step in which a linearized least-squares optimization problem is solved for the robot position. We now proceed to describe both these steps in further detail.

7.2.3.1 Orientation Estimation

Robot orientation $\theta \in (-\pi, \pi]$, thus as the robot navigates, the relative orientation measurements do not provide information about the angle wrap around. Let $\delta\theta_{i,j}$ to be the relative orientation measurement from pose x_i to x_j, then

$$\delta\theta_{i,j} = \phi(\theta_j - \theta_i), \tag{7.5}$$

where ϕ is the module operator such that $\phi(\theta) \in (-\pi, \pi]$. Thus the regularized relative orientation measurement $\delta\bar{\theta}$ is

$$\overline{\delta\theta}_{i,j} = \theta_j - \theta_i + 2k_{ij}\pi. \tag{7.6}$$

Here k_{ij} is the integer ambiguity. In [24], the authors present an integer ambiguity approximation approach which exploits the fact that relative orientation measurements over a cycle in the graph add up to 0. In our approach, since absolute orientation measurements are available, the integer ambiguity can be simply be calculated as

$$k_{ij} = \texttt{round}((\overline{\delta\theta}_{i,j} - (\bar{\theta}_j - \bar{\theta}_i))/2\pi). \tag{7.7}$$

Once the regularization constants are computed, we formulate a linear estimation problem by stacking together the absolute orientation measurement vector $\bar{\boldsymbol{\theta}}$ and regularized relative orientation measurement vector $\overline{\boldsymbol{\delta\theta}}$ as

$$\boldsymbol{\beta} = \begin{bmatrix} \overline{\boldsymbol{\delta\theta}} \\ \bar{\boldsymbol{\theta}} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{B}' & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{B}} \boldsymbol{\theta} + \begin{bmatrix} \mathbf{v}_{\delta\theta} \\ \mathbf{v}_{\bar{\theta}} \end{bmatrix}, \tag{7.8}$$

which can be solved for the global orientation estimate as

$$\hat{\boldsymbol{\theta}} = (\mathbf{B}^T \mathbf{R}_{\boldsymbol{\beta}}^{-1} \mathbf{B})^{-1} \mathbf{B}^T \mathbf{R}_{\boldsymbol{\beta}}^{-1} \boldsymbol{\beta}, \tag{7.9}$$

and the estimate error covariance is $\boldsymbol{\Sigma}_{\boldsymbol{\theta}} = (\mathbf{B}^T \mathbf{R}_{\boldsymbol{\beta}}^{-1} \mathbf{B})^{-1}$.

7.2.3.2 Position Estimation

Position estimation in this problem is nearly identical to the method presented in Section 6.2.3, a key difference being that here we do not need to map any feature positions explicitly. Once a global orientation estimate $\hat{\boldsymbol{\theta}}$ is computed, we proceed to compute robot position at each pose. From Eq. 7.1 we know that a relative pose measurement from pose x_i to x_j contains a relative position measurement $\boldsymbol{\Delta}$ as

$$^{l}\mathbf{\Delta}_{ij} = \mathbf{R}_i(\mathbf{p}_j - \mathbf{p}_i), \tag{7.10}$$

where $^{l}\mathbf{\Delta}_{ij}$ is the displacement measured in the local frame of pose x_i and $\mathbf{p}_i, \mathbf{p}_j$ are the 2D positions. Abusing notation slightly, let $^{l}\hat{\mathbf{\Delta}} \sim \mathcal{N}(^{l}\mathbf{\Delta}, ^{l}\mathbf{R}_{\mathbf{\Delta}})$ be the vector of all local relative position measurements. The vector of local relative measurements $^{l}\hat{\mathbf{\Delta}}$ can be transformed to the global frame similar to Eq. 7.2. Thus replacing $\mathbf{R}_i$ with $\mathbf{R}(\hat{\theta}_i)$, and using Eq. 7.2, we can formulate the linear estimation problem as

$$^{w}\hat{\mathbf{\Delta}} = \hat{\mathbf{R}}\ ^{l}\hat{\mathbf{\Delta}} = \mathbf{A}'\mathbf{p} + ^{w}\mathbf{v}_{\mathbf{\Delta}}. \tag{7.11}$$

$\hat{\mathbf{R}} = \mathbf{R}(\hat{\boldsymbol{\theta}})$ is the corresponding composition of rotation matrices parametrized by the estimated heading $\hat{\boldsymbol{\theta}}$, $\mathbf{p}$ is the vector of robot positions, $\mathbf{A}'$ is a matrix with each row containing elements of the set $\{-1, 0, +1\}$ and $^{w}\mathbf{v}_{\mathbf{\Delta}} \sim \mathcal{N}(\mathbf{0}, ^{w}\mathbf{R}_{\mathbf{\Delta}} = \mathbf{C}^{Tl}\mathbf{R}_{\mathbf{\Delta}}\mathbf{C})$ is the noise vector.

From Section 7.2.3.1, we know that due to correlated heading estimates, solving Eq. 7.11 directly would result in an erroneous estimate as the correlations have not been accounted for yet. After computing the orientation estimates $\hat{\boldsymbol{\theta}}$ along with the transformed global relative position measurements we stack them to give us a new measurement vector $\boldsymbol{\gamma}$. Then we have

$$\boldsymbol{\gamma} = \mathbf{h}_w(^{l}\mathbf{\Delta}, \boldsymbol{\theta}) + \mathbf{v}_w = \begin{bmatrix} \hat{\mathbf{R}}\ ^{l}\hat{\mathbf{\Delta}} \\ \hat{\boldsymbol{\theta}} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}' & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \mathbf{p} \\ \boldsymbol{\theta} \end{bmatrix} + \begin{bmatrix} ^{w}\mathbf{v}_{\mathbf{\Delta}} \\ \mathbf{v}_{\boldsymbol{\theta}} \end{bmatrix}. \tag{7.12}$$

The error covariance $\mathbf{R}_{\boldsymbol{\gamma}}$ of measurement vector $\boldsymbol{\gamma}$ is then given by,

$$\mathbf{R}_{\boldsymbol{\gamma}} = \bar{\nabla}\mathbf{h}_w \begin{bmatrix} {}^{l}\mathbf{R}_{\boldsymbol{\Delta}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \bar{\nabla}^T \mathbf{h}_w \tag{7.13}$$

where $\bar{\nabla}\mathbf{h}_w$ is the Jacobian of measurement function $\mathbf{h}_w$ (Eq. 7.12) given by

$$\bar{\nabla}\mathbf{h}_w = \begin{bmatrix} \mathbf{R} & \mathbf{M}\,{}^{l}\hat{\boldsymbol{\Delta}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{7.14}$$

where $\mathbf{M} = \frac{\partial \mathbf{C}^T}{\partial \boldsymbol{\theta}}$. Thus we have

$$\mathbf{R}_{\boldsymbol{\gamma}} = \begin{bmatrix} {}^{w}\mathbf{R}_{\boldsymbol{\Delta}} + \mathbf{M}\boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{M}^T & \mathbf{M}\boldsymbol{\Sigma}_{\boldsymbol{\theta}} \\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{M}^T & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix}. \tag{7.15}$$

Finally, the solution to the linear estimation problem of Eq. 7.12 is given by

$$\begin{bmatrix} \mathbf{p}^* \\ \boldsymbol{\theta}^* \end{bmatrix} = (\mathbf{A}^T \mathbf{R}_{\boldsymbol{\gamma}}^{-1} \mathbf{A})^{-1} \mathbf{A}^T \mathbf{R}_{\boldsymbol{\gamma}}^{-1} \boldsymbol{\gamma}. \tag{7.16}$$

Note that the above equation involves the inversion of a large sparse matrix $\mathbf{R}_{\boldsymbol{\gamma}}$. We use the same trick as in Eq. 6.17 to analytically compute the information matrix.

7.2.4 Analysis

Fusing orientation information is the key to adding robustness in the front-end and back-end. A robust front-end with accurate localization is able to make robust data associations for correlative scan matching in both local submaps and for global loop closure detection. Small errors in relative orientation estimates add up over time to create rapid growth in position error when unchecked with absolute orientation measurements. This problem arises due to the non-linear nature (trigonometric functions) of orientation. It is one of the key reasons why practical SLAM applications

often require a human to manually fix maps and correct loop closure constraints. We now proceed to first present an analysis of growth in position error due noisy relative orientation measurements followed by a discussion the accuracy of orientation estimation itself.

Let $\mathbf{x}_0 = [0, 0, 0]^T$ be the pose of the robot at time t_0 which is known. Let ${}^l\mathbf{\Delta}_{ij}$ be the local relative position measurement and $\delta\hat{\theta}_{ij}$ to be the relative orientation measurement between poses x_i and x_j. At each relative orientation measurement, let $\delta\tilde{\theta}_{ij}$ be a small error, hence

$$\delta\hat{\theta}_{ij} = \delta\theta_{ij} + \delta\tilde{\theta}_{ij}. \tag{7.17}$$

Then, 2D position $\mathbf{p}_n$ of pose x_n is given by,

$$\mathbf{p}_n = \mathbf{x}_0 + \sum_{i=0}^{n-1} {}^w\mathbf{\Delta}_{i,i+1}. \tag{7.18}$$

The above equation can be expanded as

$$\mathbf{p}_n = \mathbf{x}_0 + {}^l\mathbf{\Delta}_{01} + \ldots + (\prod_{i=0}^{i=n-2} \mathbf{R}(\delta\theta_{i,i+1}))\ {}^l\mathbf{\Delta}_{k-1,k}. \tag{7.19}$$

We can condense the above equation as,

$$\mathbf{p}_n = \mathbf{x}_0 + \sum_{k=1}^{n} \big[(\prod_{i=0}^{i=k-2} \mathbf{R}(\delta\theta_{i,i+1}))\ {}^l\mathbf{\Delta}_{k-1,k}\big]. \tag{7.20}$$

7.2.4.1 Noisy Relative Orientation Measurements

We proceed to analyze the case when only relative orientation measurements are noisy. For clarity of presentation and without loss of generality; (i) we drop the pose subscript and use the fact that x_0 is the origin, (ii) we assume that relative rotation

at each step for the true motion of the robot is fixed, thus we fix $\mathbf{R}(\delta\theta_{i,i+1}))$ as $\mathbf{R}(\delta\theta))$ and (iii) the relative linear displacement ${}^{l}\mathbf{\Delta}_{n-1,n}$ is fixed to $\mathbf{\Delta}$ and is known perfectly. Then using Eq. 7.20 we have

$$\mathbf{p}_n = \sum_{k=1}^{n} (\mathbf{R}(\delta\theta))^{k-1}\mathbf{\Delta}. \tag{7.21}$$

Using the above equation, assuming that the only measurement error is in the relative orientation information, the estimated 2D position $\hat{\mathbf{p}}_n$ is given by,

$$\begin{aligned} \hat{\mathbf{p}}_n = & \quad \textstyle\sum_{k=1}^{n} (\mathbf{R}(\delta\hat{\theta}))^{k-1}\mathbf{\Delta} \\ = & \quad \textstyle\sum_{k=1}^{n} (\mathbf{R}(\delta\theta)\mathbf{R}(\delta\tilde{\theta}))^{k-1}\mathbf{\Delta}. \end{aligned} \tag{7.22}$$

Then the error $\mathbf{e_p}$ at pose x_n is

$$\mathbf{e}_{p_n} = \sum_{k=1}^{n} ((\mathbf{R}(\delta\theta)\mathbf{R}(\delta\tilde{\theta}))^{k-1} - (\mathbf{R}(\delta\theta))^{k-1})\mathbf{\Delta}. \tag{7.23}$$

Setting $\mathbf{R}(\delta\theta) = \mathbf{I}$ without loss of generality, we have

$$\mathbf{e}_{p_n} = \sum_{k=1}^{n} ((\mathbf{R}(\delta\tilde{\theta}))^{k-1} - \mathbf{I})\mathbf{\Delta}. \tag{7.24}$$

Assuming small noise, i.e., small $\delta\tilde{\theta}$, and using $cos(\delta\tilde{\theta}) \approx 1, sin(\delta\tilde{\theta}) \approx \delta\tilde{\theta}$ we can write $\mathbf{R}(\delta\tilde{\theta})$ as

$$\mathbf{R}(\delta\tilde{\theta}) = \begin{bmatrix} cos(\delta\tilde{\theta}) & -sin(\delta\tilde{\theta}) \\ sin(\delta\tilde{\theta}) & cos(\delta\tilde{\theta}) \end{bmatrix} = \begin{bmatrix} 1 & -\delta\tilde{\theta} \\ \delta\tilde{\theta} & 1 \end{bmatrix}. \tag{7.25}$$

We have

$$\mathbf{R}(\delta\tilde{\theta})^2 = \begin{bmatrix} 1-\delta\theta^2 & -2\delta\tilde{\theta} \\ 2\delta\tilde{\theta} & 1-\delta\theta^2 \end{bmatrix} \approx \begin{bmatrix} 1 & -2\delta\tilde{\theta} \\ 2\delta\tilde{\theta} & 1 \end{bmatrix}. \tag{7.26}$$

Generalizing the above result,

$$\mathbf{R}(\delta\tilde{\theta})^k \approx \begin{bmatrix} 1 & -k\delta\tilde{\theta} \\ k\delta\tilde{\theta} & 1 \end{bmatrix}. \tag{7.27}$$

Thus using the above equation and Eq. 7.24, we get

$$\mathbf{e}_{p_n} = \sum_{k=1}^{n} \begin{bmatrix} 0 & -(k-1)\delta\tilde{\theta} \\ (k-1)\delta\tilde{\theta} & 0 \end{bmatrix} \mathbf{\Delta} \tag{7.28}$$

Setting $\mathbf{\Delta} = [1,1]^T$, and using the fact that sum of first n natural numbers is $n(n+1)/2$, we get

$$\mathbf{e}_{p_n} = \begin{bmatrix} -\frac{(n-1)(n-2)}{2}\delta\tilde{\theta} \\ \frac{(n-1)(n-2)}{2}\delta\tilde{\theta} \end{bmatrix} \mathbf{\Delta}. \tag{7.29}$$

Thus the error in position grows quadratically as the robot moves, Fig. 7.4(b) confirms this with empirical results. For zero-mean Gaussian noise we have, using Eq. 7.24 (setting $\mathbf{R}(\delta\tilde{\theta}) = \tilde{\mathbf{R}}$)

$$\mathbb{E}[\mathbf{e}_{p_n}] = \sum_{k=1}^{n}((\mathbb{E}[\tilde{\mathbf{R}}])^{k-1} - \mathbf{I})\mathbf{\Delta} \tag{7.30}$$

$$= \mathbf{0} \tag{7.31}$$

and

Coefficient	Value	95% confidence bounds
p_1	37.62	(17.41, 57.83)
p_2	510.4	(272.1, 748.7)
p_3	1750	(1060, 2440)

Table 7.1: Coefficients (with 95% confidence bounds) for Eq. 7.33.

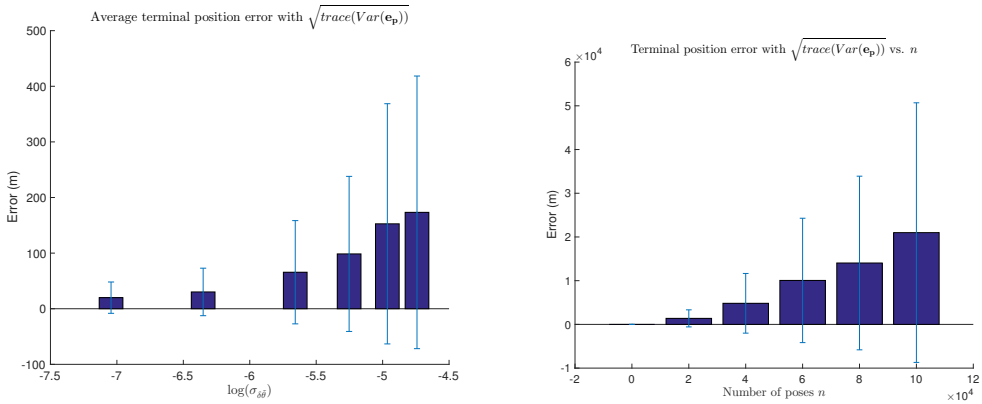
$$Var(\mathbf{e}_{p_n}) = \mathbf{e}_{p_n}^T \mathbf{e}_{p_n}. \tag{7.32}$$

For $\mathbf{\Delta} = [1, 1]^T$, Fig. 7.4(a) shows the terminal position error for $n = 1000$ poses (1000m trajectory) averaged over 50 runs for varying relative orientation noise values. It clearly shows the super-linear growth in position error for small relative heading measurement errors. The position error fits a quadratic polynomial (using MATLAB fit function) with the coefficient values in Table 7.1. Figure 7.4(b) shows the variation in terminal position error as the trajectory length is increased from 1m to 100km for a small heading error standard deviation of 0.05°. We draw attention to the fact that position error grows super-linearly (quadratic) as the robot moves and the $\sqrt{trace(Var(\mathbf{e_p}))}$ bound is $\approx 50\%$ of the distance traveled at 100km. Thus, if heading estimates are not fixed with accurate global measurements, the SLAM-front end is bound to drift and have large uncertainty such that it may not be able to detect a loop closure.

$$f_{e_y} = p_1(log(\delta\tilde{\theta}))^2 + p_2(log(\delta\tilde{\theta})) + p_3 \tag{7.33}$$

7.2.4.2 Noisy Relative Position Measurements

We proceed to analyze the case when only relative position measurements are noisy. Using Eq. 7.20 and for clarity of presentation and without loss of generality;



(a) (b)

Figure 7.4: Numerical results for analysis in Section 7.2.4 for 50 monte carlo simulations with only relative orientation measurement noise. (a) Average terminal position error with $\sqrt{trace(Var(\mathbf{e_p}))}$ bounds after 100 steps as $\sigma_{\delta\tilde{\theta}}$ varies from $[0.05°, 0.1°, 0.2°, 0.3°, 0.4°, 0.5°]$. The x-axis is plotted on a logarithmic scale for clear visualization. Note the super-linear growth in position error as relative orientation noise is stepped up. (b) Average terminal position error with $\sqrt{trace(Var(\mathbf{e_p}))}$ bounds for heading noise $\sigma_{\delta\tilde{\theta}} = 0.05°$. The trajectory length was varied from 1m to 100km. Note that at 100km the $\sqrt{trace(Var(\mathbf{e_p}))}$ bound is $\approx$ 50km which is 50% of the distance traveled.

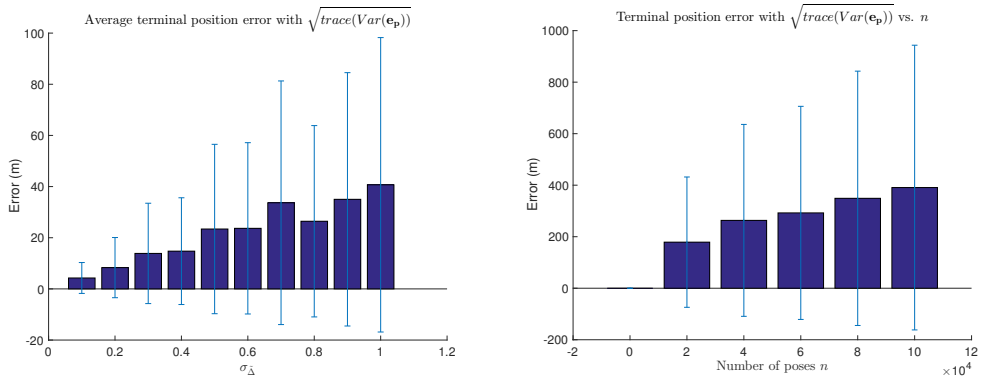
(i) we drop the pose subscript and use the fact that x_0 is the origin, (ii) we assume that relative rotation at each step for the true motion of the robot is perfectly known thus we fix $\mathbf{R}(\delta\theta_{i,i+1})$ as $\mathbf{R}(\delta\theta) = I$. Then we have

$$\mathbf{p}_n = \sum_{k=1}^{n} \mathbf{\Delta}_{k-1,k}. \tag{7.34}$$

With noisy measurements we have

$$\hat{\mathbf{p}}_n = \sum_{k=1}^{n} (\mathbf{\Delta}_{k-1,k} + \tilde{\mathbf{\Delta}}_{k-1,k}), \tag{7.35}$$

where $\tilde{\mathbf{\Delta}}_{k-1,k}$ is the zero-mean Gaussian noise in relative position measurements. The position estimation error $\mathbf{e_p}$ at the n-th pose is



(a) (b)

Figure 7.5: Numerical results for analysis in Section 7.2.4 for 50 monte carlo simulations with only relative position measurement noise. (a) Average terminal position error with $\sqrt{trace(Var(\mathbf{e_p}))}$ bounds after 100 steps as $\sigma_{\tilde{\Delta}_x}, \sigma_{\tilde{\Delta}_y}$ varies from 0.1m to 1.0m in steps of 0.1m. (b) Average terminal position error with $\sqrt{trace(Var(\mathbf{e_p}))}$ bounds for heading noise $\sigma_{\tilde{\Delta}_x} = \sigma_{\tilde{\Delta}_y} = 1$m. The trajectory length was varied from 1m to 100km. Note that at 100km the $\sqrt{trace(Var(\mathbf{e_p}))}$ bound is ≈ 1000m which is 1% of the distance traveled.

$$\mathbf{e}_n = \sum_{k=1}^{n} (\tilde{\mathbf{\Delta}}_{k-1,k}). \tag{7.36}$$

Setting $\tilde{\mathbf{\Delta}}_{k-1,k} = \tilde{\mathbf{\Delta}}$, we can re-write the above equation as

$$\mathbf{e}_n = n\tilde{\mathbf{\Delta}}. \tag{7.37}$$

Thus the error in position grows linearly as the robot moves. We contrast this with the case where we have only relative orientation noise wherein position error grows quadratically. From a probabilistic perspective, we have the expected error

$$\mathbb{E}[\mathbf{e}_n] = \mathbf{0}. \tag{7.38}$$

The variance in the error,

$$Var[\mathbf{e}_n] = \mathbf{e}_n^T \mathbf{e}_n. \tag{7.39}$$

Eq. 7.36 shows that position error is linear in the relative position error measurement.

Figure 7.5(a) shows the terminal position error for $n = 1000$ poses (1000m trajectory) averaged over 50 runs for varying relative position measurement noise values. Compared to the case wherein we varied only relative orientation noise, there is a marked difference in the behavior of terminal position error as shown in Fig. 7.4(a). The primary difference being that with increasing relative orientation noise, the terminal position error grows monotonically in a quadratic fashion whereas for only noisy relative position the error fluctuates. Figure 7.5(b) shows the variation in terminal position error as the trajectory length is increased from 1m to 100km for a relatively high relative position error (standard deviation of position error is 1m which is equivalent one-step motion of the robot). **We draw attention to the fact that position error growth is linear with noisy relative position measurements whereas it is quadratic with noisy relative orientation measurements.** Note that the $\sqrt{trace(Var(\mathbf{e_p}))}$ bound is 1% of the distance traveled at 100km whereas with small relative orientation noise of 0.05° the same bound is $\approx 50\%$ of the distance moved. This clearly indicates that; (i) the cumulative effect of small deviations in relative orientation measurements results in unreliable localization for long-term autonomy and (ii) localization uncertainty has the capability to render data-associations for global loop closure detection quite unreliable. Unless absolute orientation measurements are available to correct drift before loop closure, long-term autonomous navigation is a hard problem to tackle.

7.2.4.3 Noisy Relative Position and Orientation Measurements

We proceed to analyze the general case when both relative orientation and position measurements are noisy. For clarity of presentation and without loss of generality; (i) we drop the pose subscript and use the fact that x_0 is the origin, (ii) we assume that relative rotation at each step for the true motion of the robot is fixed, thus we fix $\mathbf{R}(\delta\theta_{i,i+1})$ as $\mathbf{R}(\delta\theta)$ and (iii) the relative linear displacement ${}^{l}\mathbf{\Delta}_{n-1,n}$ is fixed to $\mathbf{\Delta}$ and is known perfectly. Then using Eq. 7.20 we have

$$\mathbf{p}_n = \sum_{k=1}^{n} (\mathbf{R}(\delta\theta))^{k-1}\mathbf{\Delta}. \tag{7.40}$$

Using the above equation, the estimated 2D position $\hat{\mathbf{p}}_n$ is given by,

$$\begin{aligned} \hat{\mathbf{p}}_n = & \quad \textstyle\sum_{k=1}^{n} (\mathbf{R}(\delta\hat{\theta}))^{k-1}\hat{\mathbf{\Delta}} \\ = & \quad \textstyle\sum_{k=1}^{n} (\mathbf{R}(\delta\theta)\mathbf{R}(\delta\tilde{\theta}))^{k-1}[\mathbf{\Delta} + \tilde{\mathbf{\Delta}}]. \end{aligned} \tag{7.41}$$

Substituting $\mathbf{R} = \mathbf{R}(\delta\theta)$ and $\tilde{\mathbf{R}} = \mathbf{R}(\delta\tilde{\theta})$, and using $\mathbf{e}_n = \hat{\mathbf{p}}_n - \mathbf{p}_n$ the error in position can then be written as

$$\mathbf{e}_n = \big[\sum_{k=1}^{n} \mathbf{R}^{k-1}(\tilde{\mathbf{R}}^{k-1} - \mathbf{I})\big]\mathbf{\Delta} + \big[\sum_{k=1}^{n} (\mathbf{R}\tilde{\mathbf{R}})^{k-1}\big]\tilde{\mathbf{\Delta}}. \tag{7.42}$$

Using knowledge of Eq. 7.28 we can write the above result as,

$$\mathbf{e}_n = \big[\sum_{k=1}^{n} \mathbf{R}^{k-1} \begin{bmatrix} 0 & -(k-1)\delta\tilde{\theta} \\ (k-1)\delta\tilde{\theta} & 0 \end{bmatrix}\big]\mathbf{\Delta} + \big[\sum_{k=1}^{n} (\mathbf{R})^{k-1} \begin{bmatrix} 1 & -(k-1)\delta\tilde{\theta} \\ (k-1)\delta\tilde{\theta} & 1 \end{bmatrix}\big]\tilde{\mathbf{\Delta}}. \tag{7.43}$$

Thus position error is quadratic in the number of poses (i.e., distance moved) when relative orientation noise $\delta\tilde{\theta}$ is present and linear in the relative position noise $\tilde{\mathbf{\Delta}}$.

7.2.4.4 Noisy Relative Position with Absolute Orientation Measurements

We now show that with absolute orientation sensing, the compounding growth in localization error due to noisy relative orientation measurements can be easily avoided. We proceed to analyze the case when only position measurements are noisy and unbiased zero-mean Gaussian noise absolute orientation estimates are available. For clarity of presentation and without loss of generality; (i) we drop the pose subscript and use the fact that x_0 is the origin, (ii) we assume that relative linear displacement ${}^{l}\mathbf{\Delta}_{i,i+1}$ is fixed to $\mathbf{\Delta}$ and is known perfectly. Then using Eq. 7.20 we have the estimated pose

$$\hat{\mathbf{p}}_n = \sum_{k=1}^{n} \big[\mathbf{R}(\bar{\theta}_k)\ \mathbf{\Delta}\big], \tag{7.44}$$

where $\bar{\theta}_k \sim \mathcal{N}(\theta_k, \sigma^2_{\bar{\theta}_k})$ is the absolute orientation measurement at time t_k. Each absolute orientation measurement is independent thus there is no multiplicative effect of noisy relative orientation measurements. Thus position error is given by

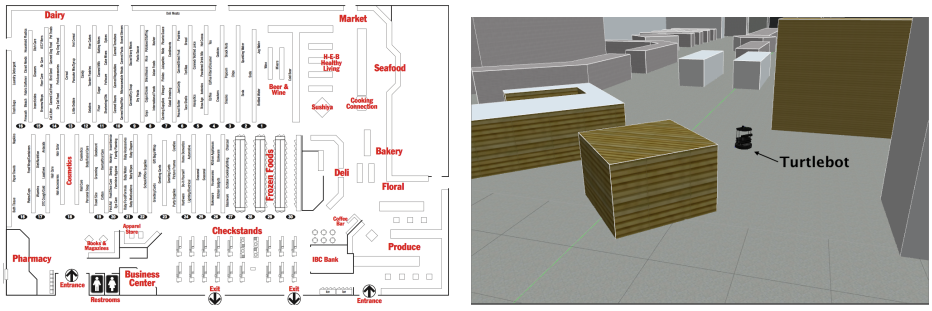
$$\mathbf{e}_n = \big[\sum_{k=1}^{n}(\mathbf{R}(\bar{\theta}_k) - \mathbf{R}(\theta_k))\big]\mathbf{\Delta} + \big[\sum_{k=1}^{n}\mathbf{R}(\bar{\theta}_k)\big]\tilde{\mathbf{\Delta}}. \tag{7.45}$$

We can see from the above equation that position error grows linearly in the distance moved.

7.3 Simulation Results

We tested our approach in multiple high fidelity simulations. To simulate a realistic scenario we obtained the floor plan (shown in Fig. 7.6(a)) for a large local

HEB grocery store with a 78,000 square feet footprint with a 36 foot high ceiling in College Station, Texas. Using the floor plan we constructed a virtual store environment in 3D and drove a turtlebot equipped with a 2D Lidar, gyro, wheel encoders and a ceiling facing camera (orientation sensing) using Gazebo in ROS. Figure 7.6(b) shows a virtual turtlebot operation in the simulated environment. To generate the simulation results we created a series of waypoints through the building which the robot must follow during each Monte Carlo run.



(a) Unscaled floor plan of the physical building.

(b) A virtual turtlebot operating in a grocery store environment modeled in Gazebo.

Figure 7.6: (a) Shows the unscaled floor plan for a local grocery store which was used to model the virtual environment. Building scale was estimated using Google earth. (b) Shows an instance of a virtual turtlebot operating in the simulated environment.

7.3.1 Heading Estimation

Figure 7.7 shows pictures taken at the real-world facility. The ceiling has distinctly visible corrugation patterns and structured light fixtures. The ceiling texture of the virtual grocery store replicates the corrugation pattern observed in the physical building. A ceiling facing camera is used to detect line features arising from the corrugated roofing. We use the heading estimation method as described in Sec-

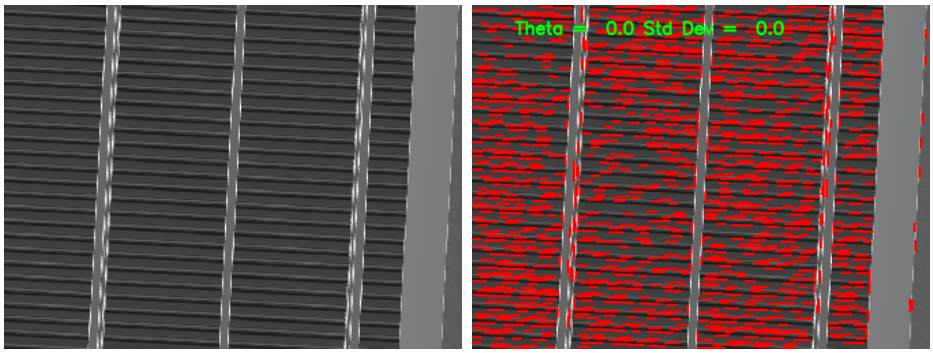
(a) (b) (c)

Figure 7.7: The ceiling of the actual physical grocery store on which the virtual environment was modeled. Note that in (a) and (b) the ceiling corrugation pattern is strongly visible. In (c) we can see that the rows of lights hanging from the ceiling also follow a strict pattern which may be detected and easily used as an orientation feature.

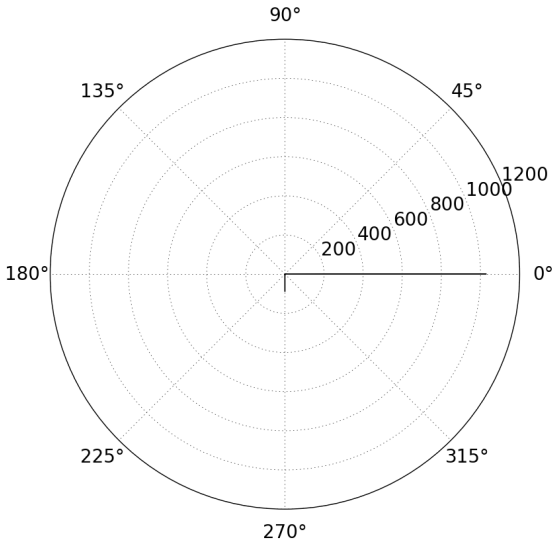
tion 7.2.1. Figure 7.8 shows images of the simulated ceiling and a polar plot of the ceiling direction histogram. We use a bin size $b = 0.1^\circ$ for the histogram operation described in Section 7.2.1. Fig. 7.9 shows the heading estimation error over a 400s drive through the simulated grocery store environment.

7.3.2 Mapping and Localization

Table 7.2 shows the RMS localization error for a total of 200 runs with two different LiDAR ranges of 20m and 30m. With our approach the average RMS position error was 1 order of magnitude smaller than the standard pose-graph SLAM approach. The results in Table 7.2 show that while the standard approach suffered a catastrophic failure $40\% - 50\%$ of the times **our approach never resulted in a failure**. Fig. 7.2 shows a comparison of maps created for 1 particular run. As can be seen from these results, the proposed approach constitutes a significant robustness improvement in mapping over the current state of the art. Figure 7.10 shows mapping failure and localization error for one simulation run with existing state-of-the-art

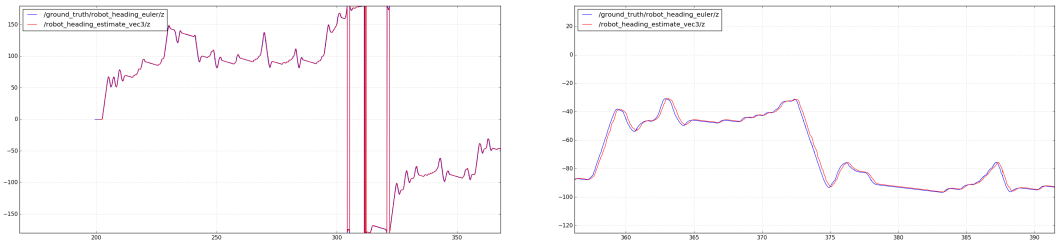


(a) (b)



(c)

Figure 7.8: The ceiling direction estimation process for an image captured from the ceiling camera in the simulation study. (a) Shows a view of the ceiling in the simulated grocery store model. It replicates the corrugation pattern present on typical industrial building roofings. In (c) we see results of the line detection algorithm (red) and the calculated ceiling direction (green). In (d) the ceiling direction measurements are plotted on polar histogram, the plot shows a strong response for "major" direction of the ceiling, i.e., building North-South. Note that in the histogram shows that lines were detected in two directions (perpendicular to each other), this is due to the fact that lines were detected from secondary features (beams etc.), i.e., not the ceiling corrugation as shown in (b).



(a) (b)

Figure 7.9: Heading estimate (red) and the true value (red) for a simulation run in the grocery store environment. (a) Shows the heading estimate and ground truth for a 200s period. In (b) we zoom into a $\approx$ 40s window. Note that heading estimates lag behind the ground truth, this can be attributed to the orientation measurement update rate $\approx$ 30Hz.

methods. The scan matching algorithm detects an incorrect loop closure due to drift in localization which corrupts the map and the localization diverges rapidly from the ground truth. This error is representative of the most commonly occurring reason for mapping failure in scan matching based techniques.

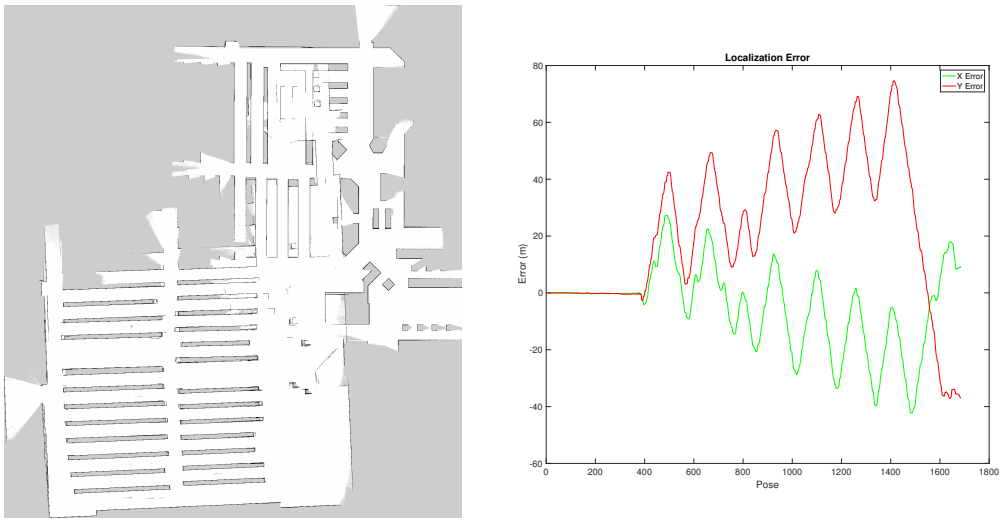
7.4 Physical Experiment Results

We conducted physical experiments with a retrofitted turtlebot to demonstrate the reliability of our mapping algorithm in a warehouse, which is a commonly occurring industrial robotics environment. The facility used was Texas A&M University's Surplus warehouse located at 957 Agronomy Rd, College Station, Texas (see Fig. 7.11(a)). The facility has a total floor area of $66,000$ sq. ft. The warehouse is a dynamic environment with constantly moving people, vehicles and equipment.

7.4.1 Robot Setup

The robot used in our experiments is shown in Fig. 7.12. It is equipped with easily available low-cost commercial sensors:

1. A 360° LiDAR (RP Lidar) with a range of 5m.



(a) (b)

Figure 7.10: Simulation results for one experiment with 30m range. (a) Shows the mapping failure that results in faulty loop closure detection. (b) Shows the localization error at each pose. Note that around pose 400, loop closure makes a faulty estimation which results in a catastrophic failure.

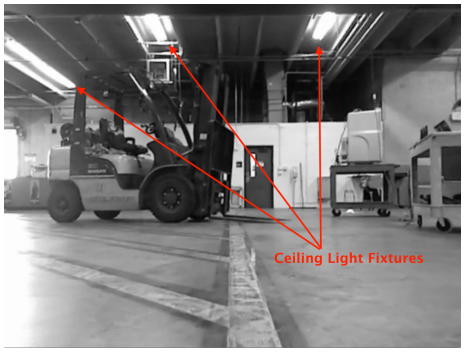
2. A monocular ceiling facing camera (Pt Grey Chameleon) is used. The camera runs at 30Hz, with an image resolution of 640×480.

3. A Razor 9-DOF IMU, of which only the yaw rate gyro is used.

4. A commercial off-the-shelf laptop (Dell Latitude E5430, Core-i7 2.4 GHz, 12 GB RAM) for computational purposes and data logging purposes.

The objective with this robot setup was to show that we are able to achieve highly reliable mapping using low-cost commercial sensors for large scale environments. We drove the robot around the facility and used ROS bags to store the data. The data was then post-processed using our approach and state-of-the-art. Thus identical sensor data was used for both estimates.

(a) The building measures $95m \times 64m$ and is a dynamic environment with constantly moving people, vehicles (vans, forklifts etc.) and goods.



(b) Robot's view from its forward facing camera. The ceiling has clearly visible lighting fixtures which are used as orientation information sources.

Figure 7.11: (a) Google Earth view of Texas A&M University's surplus warehouse facility. (b) Front-view from the robot's camera at the start of the mapping run.

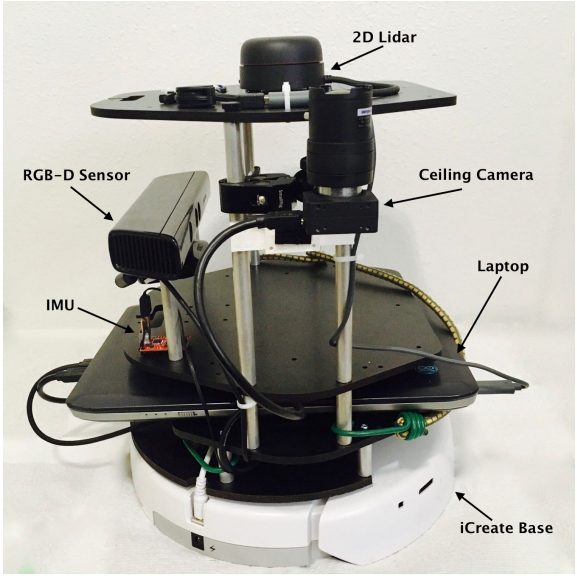


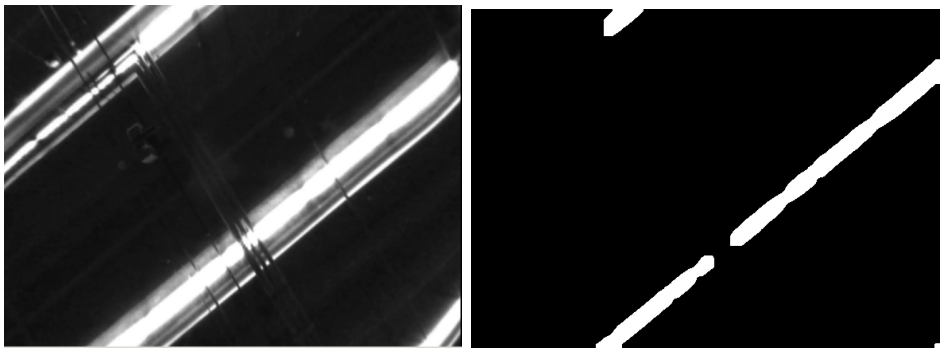
Figure 7.12: Side-view of the physical robot used in real-world warehouse experiment.

7.4.2 Heading Estimation

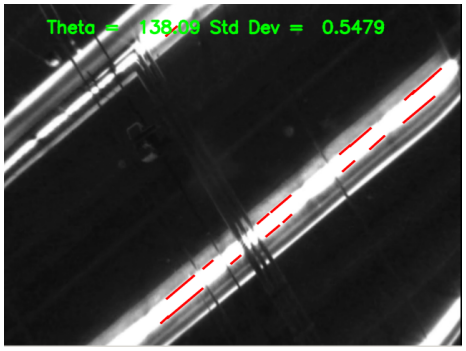
An initial survey of the warehouse facility revealed that the ceiling was equipped with rectangular light fixtures at regular intervals which we decided to leverage for orientation estimation. For the purposes of our experiment, we threshold the image such that we get a binary image, thus only ceiling lights appear as rectangular bright spots while rest of the image appears black. After thresholding we follow the process described in Algorithm 4. Fig. 7.13 shows the ceiling camera's view and the thresholded binary image along with heading detection. Heading estimates were computed at 30 Hz. We use a bin size $b = 0.1^\circ$ for the histogram operation described in Section 7.2.1.

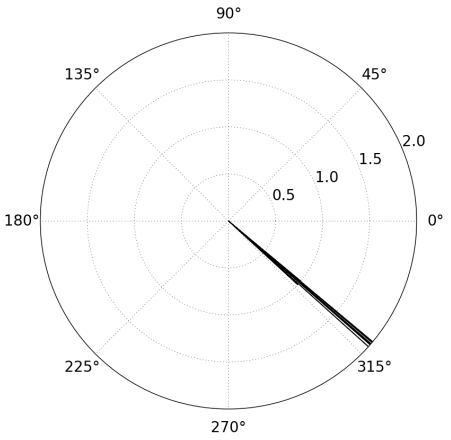
7.4.2.1 Orientation Estimation Accuracy

Assuming no inherent bias, heading accuracy is dependent on the number of feature measurements available from the ceiling. More measurements result in better estimation (Law of Large Numbers), this can be seen in the contrast of the polar plots in Fig. 7.8(c) for the simulation where we observed ≈ 1000 features in the heaviest bin vs. Fig. 7.13(d) where we observed 2 features in the heaviest bin for the real world experiment. In the simulation we are able to achieve orientation estimation with a standard deviation of $\approx< 0.1^\circ$ whereas for the real-world case the number is $\approx 0.5^\circ$. Figure 7.14(c) shows that when the simulated ceiling height is lowered to 18 feet, the number of measurements drops to ≈ 100 as opposed to ≈ 1000 available when the ceiling is at a height of 36 feet (see Fig. 7.8(c)). This has the direct effect of reducing the accuracy of heading measurements.



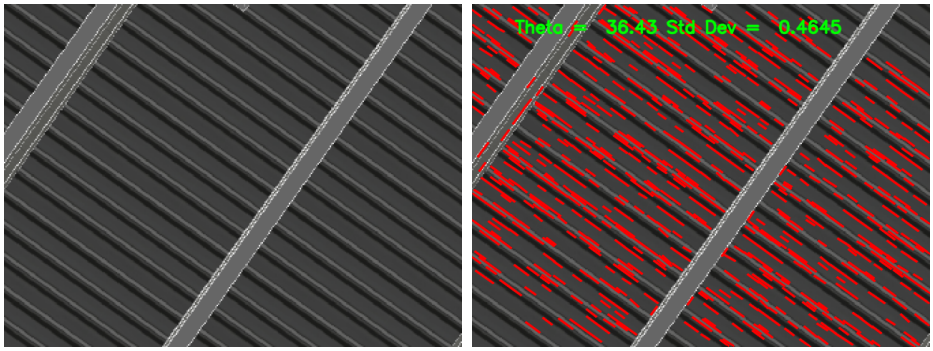
(a) (b)





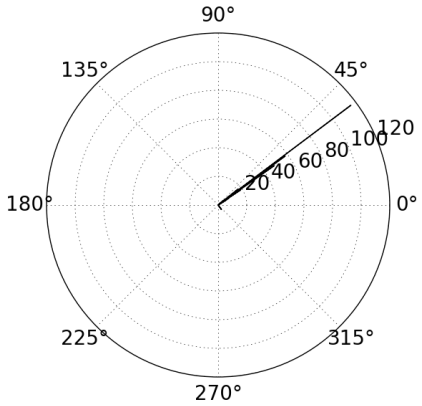
(c) Ceiling direction mean $\theta_c = 138.09°$, $\sigma_c = 0.5479°$. (d)

Figure 7.13: View of the ceiling from the upward facing camera mounted on our platform. The orientation sensing system uses straight line light fixtures to detect line features and find the principal direction of the ceiling in the image frame.



(a)

(b) Ceiling direction mean $\theta_c = 36.43°$, $\sigma_c = 0.465°$.



(c)

Figure 7.14: Heading estimation for a simulated roof with corrugation. In this simulation the ceiling is at a height of 18 ft.

7.4.3 Mapping

Fig. 7.15 shows mapping results for the physical experiment. Fig. 7.15(a) shows the map as computed by the standard correlative scan matching based SLAM approach. The map suffers a catastrophic failure characterized by a twist in the estimated geometry of the environment. Fig. 7.15(b) shows the map as estimated by our approach. Our map is globally consistent and does not suffer major twisting whereas the map estimated with the state-of-the-art approach is bent out of shape

(a) (b)

Figure 7.15: Mapping results for physical warehouse experiment. (a) Estimated map with a standard SLAM approach using correlative scan matching in the front-end and G2O in the back-end. (b) Estimated map with our approach.

and unusable for navigation in its current form. A key difference that may be noted is; the edges of environment geometry without using orientation appear sharper (Fig. 7.15(a)) in contrast to the geometry as estimated when absolute orientation sensing is used (Fig. 7.15(b)). Heading estimation noise is the driving factor behind this local *map smudging*. Thus higher accuracy in heading estimation should eliminate this issue. We note that in simulations this issue is not observed as heading estimation is more accurate (see discussion in Section 7.4.2.1).

7.5 Discussion

Our method is the first of its kind to fuse absolute orientation sensing in a 2D pose-graph formulation. Through a detailed analysis we showed when relative orientation noise is present, position error grows quadratically in the distance moved,

i.e., it has a far greater affect on mapping and localization uncertainty than relative position error. Monte Carlo simulations in the analysis demonstrated a quadratic growth in position error as heading noise is increased. The key point to note is that the localization uncertainty, i.e., the confidence bounds grow at a faster rate than localization error. It is more important to study error confidence than the error itself. With high error bounds, which also grow quadratically with growing relative orientation error, one cannot reasonably expect a high-likelihood of detecting loop closure in realistic physical environments. As an example, if a robot were to operate a in large warehouse, the scan matcher would not be able to distinguish between two aisles that look the same, it would be up to the localization prior to provide an initial guess for the loop closure search. With an erroneous initial guess it is only too well understood that a non-linear optimizer will converge to arbitrary local minima. Simulation results in a realistic physical environment confirm this finding. Through extensive Monte Carlo simulations we have shown that our approach is an order of magnitude more accurate and never fails in testing whereas existing mapping methods fail $\approx 40\% - 50\%$ of the times. Further, we confirmed the robustness of our approach in a physical experiment wherein we mapped a live warehouse facility and showed that a regular scan matching approach fails to estimate a consistent map of the environment due to localization drift which also prevents loop closure detection. Thus for long-term autonomy, it behooves us to capture global orientation cues to make SLAM robust and reliable. The heading estimation and fusion techniques developed in this work are applicable to any scan matching approach, i.e., both 2D and 3D. We now proceed to conclude this dissertation, discuss limitations of existing work and present exciting new directions of research that my utilize or extend this work.

Front-end	Back-end	Lidar Range	Total Runs	Failures	Average RMSE
Correlative Scan Matching	g2o	30	50	21	3.90m
Our Approach	LOGO	30	50	0	0.36m
Correlative Scan Matching	g2o	20	50	25	5.60m
Our Approach	LOGO	20	50	0	0.35m

Table 7.2: Comparison of existing state-of-the-art vs. our approach. Both methods use the same scan matching approach. Our approach suffered from 0 catastrophic failures.

8. CONCLUSIONS

The objective of this work in its entirety is to understand the problem of long-term autonomy. "Long-Term" has no industrial or academic definition based on quantitative metrics such as length of trajectory or size of map etc. Long-term operation for an assistive robot can be 10 years of 24×7 operation in an old-age home all the way to 10 weeks of operation on a different planet in unknown terrain. The role of "Long-Term" in this dissertation is to draw the reader's attention to the fact that for mobile robots to attain widespread everyday use, autonomy needs to become robust. Much more than what it is in the current state-of-the-art.

In Chapter 2 we presented an overview of the state-of-the-art in Simultaneous Localization and Planning (SLAP). We presented results for a rollout-based extension to the standard FIRM algorithm and showed that it outperforms or matches FIRM in time to complete tasks and success probability. We also showed that compared to local optimization-based approaches, this method is able to leverage the global FIRM cost-map and generate plans that do not get trapped in local minima. A limitation of several existing works including FIRM and its rollout-based extension is that they rely on a Gaussian belief assumption. A Gaussian belief assumption may not always work in real-world situations, a prime example being environments with a high degree of self-similarity. These present quite often in man-made environments such as offices, warehouses, factories etc. In Chapter 3, we developed a novel method called M3P for active data association to disambiguate non-Gaussian beliefs. We showed that our approach is able to guide a kidnapped robot to observe unique information in the world based on prior knowledge of the map and converge to a unimodal hypothesis. We applied the method of M3P to a physical robot operating in

a maze like environment which is initially kidnapped and uses M3P to disambiguate its global pose.

In Chapter 4 we provided an overview of existing literature in the field of Simultaneous Localization and Mapping (SLAM) and looked into the role of orientation in SLAM. Orientation estimation error as it turns out, is a key reason for faulty and unreliable estimation in SLAM due to the non-linear nature of trigonometric functions of heading that are involved in relative measurements between robot poses and from robot pose to features. Chapter 5 studied the role of orientation in SLAM in a 2D feature-based setting with absolute orientation measurements available through celestial sensors (e.g., star trackers etc.). We showed that loop closure reduces the error uncertainty by half at the feature estimates farthest from the loop, which raises the question,"Is it possible to avoid loop closure and still get reliable mapping?". The answer to that questions as shown in Chapter 5 is yes. We developed two approaches F2F and R2F. F2F captures relative measurements between features at each keyframes whereas R2F capture relative measurements between the robot and features at each time. For both methods our analysis showed that localization error growth is linear as the robot moves further away from start and the error growth rate can be controlled by planning how many measurements to take. We concluded that with absolute orientation measurements, accurate and consistent long-term localization with $<$ 1m error is possible for trajectories that are 100kms long. However, absolute orientation measurements may not always be available, thus it is important to understand how the underlying structure of the SLAM problem can be exploited for more robust estimation than existing state-of-the-art. In Chapter 6 we looked at the problem of 2D feature-based SLAM when absolute orientation measurements are not available. We developed a two-step pose-graph SLAM approach called RFM-SLAM that separates orientation and position estimation in SLAM using relative

feature measurements. First an on-manifold optimization is solved for the robot orientation followed by a linearized least squares for the robot and feature poses. We showed that the accuracy of trajectory estimates from RFM-SLAM degrades gracefully compared to existing non-linear optimization based approaches as sensor noise increases. Empirical results demonstrated that our approach avoids catastrophic failure which occurs commonly in non-linear optimization based SLAM approaches due to bad initial guesses.

In Chapter 7 we leveraged insights gained on the role of orientation in SLAM and applied them to a real-world scenario in a 2D scan matching based approach. Current state-of-the-art in scan-matching based relative pose graph SLAM relies on long-range LiDAR sensors for getting reliable and accurate relative pose measurements and loop closure detection. Quite often, in practical applications, a human-in-the-loop is required to fix loop closures, i.e., correct for false positive loop closure detections or add loop closures where they are not detected, in order to compute a reliable map. From the perspective of true long-term autonomy, this is certainly not desirable. For automatic loop closure detection, an issue that commonly comes up is that when a robot traverses a large distance before closing a loop ("large" depends on the ratio of LiDAR range to dimensions of the environment), there is sufficient localization drift such that accurate data association for loop closure is not possible. In self-similar environments such as warehouses, where there are multiple identically shaped aisles, this issue rears its ugly head quite often. Extensive analysis of the localization problem with noisy relative pose measurements showed that small noise in relative heading measurements has a far more disastrous consequence on localization uncertainty than noisy relative position measurements. The method of LOGO developed in Chapter 7 was applied to both extensive simulation studies for a robot operating in a large virtual building and to a real-world warehouse. Simula-

tion studies showed that our mapping method never failed whereas state-of-the-art SLAM approaches fail about $\approx 40\% - 50\%$ of the times. Further, real world testing validated that orientation cues can be extracted easily in indoor settings and can be used to robustly map large environments where a standard approach may easily fail.

8.1 Limitations and Future Work

A key limitation of the M3P approach (Chapter 3) is in the action selection step during replanning which can be computationally expensive for a large number of modes. Future work will look at reducing this cost and experiments will be extended to larger problems (e.g., symmetric office environments), with more complex perception models and drastic localization failures (e.g., sequential kidnappings). Finally, there may be tasks which are feasible with a multimodal distribution on the belief. Such cases present an interesting area for future motion planning research.

For the method of LFGO (Chapter 5), the goal is to nudge SLAM research to explore absolute orientation sensing, particularly the development of promising sensors such as daylight star trackers and MEMS-based gyrocompasses along with their application to robotics. Future work would investigate planning to incorporate obstacles, unknown data associations and terminal localization error constraints. For the method of RFM-SLAM (Chapter 6), future work involves implementing the software in more efficient frameworks, e.g. C++ to compare the time required to solve given problems with state-of-the-art solvers on publicly available datasets. Though the non-linear optimization problem for orientation may be susceptible to initial guess error, such an issue was not observed, perhaps the underlying nature of the orientation estimation problem is less sensitive to the initial guess. This aspect needs to be studied further to develop a deeper understanding of the SLAM problem.

A current limitation of the work in Chapter 7 is that its application has been

limited to 2D indoor mapping. Currently, we are investigating an application of orientation sensing using star trackers to autonomous vehicles for full 6 DoF localization. Our goal is to demonstrate robust outdoor autonomy without relying on any form of GPS. A major motivating factor for application to autonomous vehicles is that the system does not need to rely on external signals such as GPS which can be easily spoofed or jammed. This presents a threat to the safety and reliability of autonomous vehicles. Thus for long-term operation, autonomous vehicles need back-up navigation systems which are self-contained and robust.

An interesting question which may be raised is, "how can we design indoor environments to suit robust autonomous robot operation?". Given the opportunity to design environments from ground up, there are a number of *tricks* that may be used:

1. Beacon-based systems: Unique RFID tags may be embedded in walls and floors of buildings during construction. These known markers can be used by robots to localize themselves across large buildings. One can also place fiducial markers, though a common approach it may not be aesthetically pleasing for environments such as homes and offices. Bluetooth LE beacons are also gaining popularity for positioning due to growing smartphone popularity.

2. Artificial orientation cues: Visual or infrared patterns oriented along a global direction may be embedded in ceilings and other structures which the robot can track to maintain an accurate estimate of heading with respect to a fixed direction. Placing ceiling lights in a fixed global direction is another easy way of providing heading information. A dominant artificial magnetic flux may be generated such that robots can rely on low-cost magnetometers for heading.

3. Architectural Cues: Structural designs with unobstructed views of pillars from multiple locations are an easy way to add a vertical reference for roll and pitch

estimation in 3D navigation. In industrial buildings, naked ceiling beams can also provide easy to detect horizontal features which aid in heading estimation. One can paint North-South beams in a single color and East-West beams in another color, thus making it easy for the robot to disambiguate directions.

The prior list is by no means exhaustive, but it provides ideas which may be fruitfully explored in the future as robots grow to become a part of everyday workplace and home environments.

8.2 Closing Remarks

It is said that one must ask the right question and the answer presents itself naturally. We strongly believe that the right question to ask for robust and precise long-term localization is "how does a system estimate its orientation with respect to a fixed reference?". Once this question is answered, the solution to long-term autonomy flows naturally. Surely if mankind is able to send probes millions of miles into deep space and reliably hit asteroids, we can get autonomous mobile robots to localize reliably during their lifetime with only on-board sensing.

REFERENCES

[1] Linde robotics. `http://lindemh.com.au`.

[2] Ln-120g stellar-inertial-gps navigation.

[3] Openslam. `https://openslam.org`.

[4] Star tracker st400 and st200.

[5] Stellar-inertial navigation system.

[6] Video of M3P plan execution in simulation. https://www.youtube.com/watch?v=8kldm34Ya4I.

[7] P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *Found. Comput. Math.*, 7(3):303–330, July 2007.

[8] Saurav Agarwal, Vikram Shree, and Suman Chakravorty. Rfm-slam: Exploiting relative feature measurements to separate orientation and position estimation in slam. In *Proc. IEEE International Conference on Robotics and Automation (ICRA).*, May 29 - June 3 2017.

[9] Aliakbar Agha-mohammadi, Saurav Agarwal, Aditya Mahadevan, Suman Chakravorty, Daniel Tomkins, Jory Denny, and Nancy Amato. Robust online belief space planning in changing environments: Application to physical mobile robots. In *IEEE Int. Conf. Robot. Autom. (ICRA)*, Hong Kong, China, 2014.

[10] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[11] Ali akbar Agha-mohammadi, Saurav Agarwal, Suman Chakravorty, and Nancy Amato. Simultaneous localization and planning for physical mobile robots via enabling dynamic replanning in belief space. 2016. submitted to IEEE Transactions on Robotics, arXiv:1510.07380 [cs.RO].

[12] Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A. Ngo. Monte carlo value iteration for continuous-state POMDPs. In *WAFR*, volume 68 of *Springer Tracts in Advanced Robotics*, pages 175–191. Springer, 2010.

[13] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the ekf-slam algorithm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3562–3568, Oct 2006.

[14] Tim Bailey and Juan Nieto. Ekf-slam v2.0 by tim bailey and juan nieto. `http://www-personal.acfr.usyd.edu.au/tbailey/software/`.

[15] Dimitri Bertsekas. *Dynamic Programming and Optimal Control: 3rd Ed.* Athena Scientific, 2007.

[16] S.D. Bopardikar, B. Englot, and A. Speranzon. Robust belief roadmap: Planning under uncertain and intermittent sensing. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 6122–6129, May 2014.

[17] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 2, pages 1899–1906 vol.2, Sept 2003.

[18] N. Boumal, A. Singer, and P. A. Absil. Robust estimation of rotations from relative measurements by maximum likelihood. In *52nd IEEE Conference on Decision and Control*, Dec 2013.

[19] Nicolas Boumal, Bamdev Mishra, P.-A. Absil, and Rodolphe Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15, 2014.

[20] Chris Bowen and Ron Alterovitz. Closed-loop global motion planning for reactive execution of learned tasks. In *Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on*, pages 1754–1760. IEEE, 2014.

[21] Adam Bry and Nicholas Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *ICRA*, pages 723–730, 2011.

[22] L. Carlone and A. Censi. From angular manifolds to the integer lattice: Guaranteed orientation estimation with application to pose graph optimization. *IEEE Transactions on Robotics*, 30(2):475–492, April 2014.

[23] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert. Initialization techniques for 3d slam: A survey on rotation estimation and its use in pose graph optimization. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4597–4604, May 2015.

[24] Luca Carlone, Rosario Aragues, Jos A. Castellanos, and Basilio Bona. A fast and accurate approximation for planar pose graph optimization. *The International Journal of Robotics Research*, 33(7):965–987, 2014.

[25] Luca Carlone, Jingjing Du, Miguel Kaouk Ng, Basilio Bona, and Marina Indri. Active slam and exploration with particle filters using kullback-leibler divergence. *Journal of Intelligent & Robotic Systems*, 75(2):291–311, 2013.

[26] H. Carrillo, I. Reid, and J. A. Castellanos. On the comparison of uncertainty criteria for active slam. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2080–2087, May 2012.

[27] D. Caruso, J. Engel, and D. Cremers. Large-scale direct slam for omnidirectional cameras. In *iros*, sept 2015.

[28] J.A. Castellanos, J. Neira, and J.D. Tardos. Limits to the consistency of ekf-based slam. In *In IFAC Symposium on Intelligent Autonomous Vehicles*, 2004.

[29] S. Chakravorty and R. Scott Erwin. Information space receding horizon control. In *IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL)*, April 2011.

[30] Pratik Chaudhari, Sertac Karaman, David Hsu, and Emilio Frazzoli. Sampling-based algorithms for continuous-time POMDPs. In *the American Control Conference (ACC)*, Washington DC, 2013.

[31] J. Civera, O. G. Grasa, A. J. Davison, and J. M. M. Montiel. 1-point ransac for ekf-based structure from motion. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3498–3504, Oct 2009.

[32] James M Coughlan and Alan L Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 941–947. IEEE, 1999.

[33] M. Csorba, J. K. Uhlmann, and H. F. Durrant-Whyte. A sub-optimal algorithm for automatic map building. In *American Control Conference, 1997. Proceedings of the 1997*, volume 1, pages 537–541 vol.1, Jun 1997.

[34] A. Davison, L. Reid, N. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007.

[35] A. J. Davison and D. W. Murray. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865–880, Jul 2002.

[36] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. 2012.

[37] Frank Dellaert and Michael Kaess. Square root sam: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[38] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on*, 17(3):229–241, 2001.

[39] Leo Dorst. First order error propagation of the procrustes method for 3d attitude estimation. *IEEE transactions on pattern analysis and machine intelligence*, 27(2):221–229, 2005.

[40] Jingjing Du, L. Carlone, M. Kaouk Ng, B. Bona, and M. Indri. A comparative study on active slam and autonomous exploration with particle filters. In *Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME International Conference on*, pages 916–923, July 2011.

[41] Gregory Dudek, Kathleen Romanik, and Sue Whitesides. Localizing a robot with minimum travel. *SIAM Journal on Computing*, 27(2):583–604, 1998.

[42] Ethan Eade and Tom Drummond. Edge landmarks in monocular slam. *Image and Vision Computing*, 27(5):588–596, 2009.

[43] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, September

2014.

[44] J. Engel, J. Stueckler, and D. Cremers. Large-scale direct slam with stereo cameras. In *iros*, Sept 2015.

[45] J. Engel, J. Sturm, and D. Cremers. Camera-based navigation of a low-cost quadrocopter. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[46] R. Eustice, M. Walter, and J. Leonard. Sparse extended information filters: insights into sparsification. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3281–3288, Aug 2005.

[47] Hans Jacob S Feder, John J Leonard, and Christopher M Smith. Adaptive mobile robot navigation and mapping. *The International Journal of Robotics Research*, 18(7):650–668, 1999.

[48] J. Folkesson and H. Christensen. Graphical slam - a self-correcting map. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 383–390 Vol.1, April 2004.

[49] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Appearance-based active, monocular, dense reconstruction for micro aerial vehicles. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[50] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(34):195 – 207, 1998. Autonomous Mobile Robots.

[51] P. Furgale, J. Enright, and T. Barfoot. Sun sensor navigation for planetary rovers: Theory and field testing. *IEEE Transactions on Aerospace and Elec-*

tronic Systems, 47(3):1631–1647, July 2011.

[52] S. Garrido-Jurado, R. Muoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marn-Jimnez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280 – 2292, 2014.

[53] A. Gasparri, S. Panzieri, F. Pascucci, and G. Ulivi. A hybrid active global localisation algorithm for mobile robots. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3148–3153, April 2007.

[54] P. W. Gibbens, G. M. W. M. Dissanayake, and H. F. Durrant-Whyte. A closed form solution to the single degree of freedom simultaneous localisation and map building (slam) problem. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 1, pages 191–196 vol.1, 2000.

[55] Ruben Gomez-Ojeda, Francisco-Angel Moreno, Davide Scaramuzza, and Javier Gonzalez-Jimenez. PL-SLAM: a Stereo SLAM System through the Combination of Points and Line Segments. *arXiv preprint arXiv:1705.09479*, 2017.

[56] Xujun Guan, Xinlong Wang, Jiancheng Fang, and Shaojun Feng. An innovative high accuracy autonomous navigation method for the mars rovers. *Acta Astronautica*, 104(1):266 – 275, 2014.

[57] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision, 2nd Edition*. Cambridge University Press, 2004.

[58] R. He, E. Brunskill, and N. Roy. Efficient planning under uncertainty with macro-actions. *Journal of Artificial Intelligence Research*, 40:523–570, February 2011.

[59] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based rules for designing consistent ekf slam estimators. *The*

International Journal of Robotics Research, 29(5):502–528, 2009.

[60] S. Huang and G. Dissanayake. Convergence and consistency analysis for extended kalman filter based slam. *IEEE Transactions on Robotics*, 23(5):1036–1049, Oct 2007.

[61] Shoudong Huang, N. M. Kwok, G. Dissanayake, Q. P. Ha, and Gu Fang. Multi-step look-ahead trajectory planning in slam: Possibility and necessity. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 1091–1096, April 2005.

[62] Vadim Indelman, Luca Carlone, and Frank Dellaert. Planning in the continuous domain: a generalized belief space approach for autonomous navigation in unknown environments. *International Journal of Robotics Research (IJRR)*, 34(7):849–882, June 2015.

[63] Stanford Research Institute. Open karto. `https://github.com/ros-perception/open_karto`.

[64] P. Jensfelt and S. Kristensen. Active global localization for a mobile robot using multiple hypothesis tracking. *Robotics and Automation, IEEE Transactions on*, 17(5):748–760, Oct 2001.

[65] E. Jones and S. Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *International Journal of Robotics Research*, January 2011.

[66] Eagle S. Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, 30(4):407–430, 2011.

[67] S. J. Julier and J. K. Uhlmann. A counter example to the theory of simultaneous localization and map building. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4238–4243 vol.4, 2001.

[68] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Integrated task and motion planning in belief space. *Submitted. Draft at http://people. csail. mit. edu/lpk/papers/HPNBelDraft. pdf*, 2012.

[69] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Trans. on Robotics (TRO)*, 24(6):1365–1378, December 2008.

[70] M. Kaess, S. Williams, V. Indelman, R. Roberts, J. J. Leonard, and F. Dellaert. Concurrent filtering and smoothing. In *Information Fusion (FUSION), 2012 15th International Conference on*, pages 1300–1307, July 2012.

[71] Michael Kaess, Viorela Ila, Richard Roberts, and Frank Dellaert. *Algorithmic Foundations of Robotics IX: Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics*, chapter The Bayes Tree: An Algorithmic Foundation for Probabilistic Robot Mapping, pages 157–173. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[72] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert. isam2: Incremental smoothing and mapping using the bayes tree. *The International Journal of Robotics Research*, 31(2):216–235, 2012.

[73] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[74] Shoudong Huang Kasra Khosoussi, Gaurav S. Sukhatme and Gamini Dissanayake. Designing sparse reliable pose-graph slam: A graph-theoretic approach. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR), San Francisco, USA*, December 2016.

[75] L.E. Kavraki, M.N. Kolountzakis, and J.C. Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation*, 14:166–171, February 1998.

[76] L.E. Kavraki, P. Svestka, J.C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[77] K. Khosoussi, Shoudong Huang, and G. Dissanayake. Tree-connectivity: Evaluating the graphical structure of slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1316–1322, May 2016.

[78] K. Khosoussi, Shoudong Huang, and G. Dissanayake. Tree-connectivity: Evaluating the graphical structure of slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1316–1322, May 2016.

[79] Kasra Khosoussi, Shoudong Huang, and Gamini Dissanayake. Exploiting the separable structure of slam. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[80] Ayoung Kim and R.M. Eustice. Perception-driven navigation: Active visual slam for robotic area coverage. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3196–3203, May 2013.

[81] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Sym-*

posium on Mixed and Augmented Reality, ISMAR '07, pages 1–10, Washington, DC, USA, 2007. IEEE Computer Society.

[82] K. Konolige, G. Grisetti, R. Kmmerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient sparse pose adjustment for 2d mapping. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 22–29, Oct 2010.

[83] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, Feb 2001.

[84] R. Kummerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard. Large scale graph-based SLAM using aerial images as prior information. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[85] H. Kurniawati, T. Bandyopadhyay, and N.M. Patrikalakis. Global motion planning under uncertain motion, sensing, and environment map. *Autonomous Robots*, 33(3):255–272, 2012.

[86] H. Kurniawati, D. Hsu, and W.S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Proceedings of Robotics: Science and Systems*, 2008.

[87] H. Kurniawati and V. Yadav. An online pomdp solver for uncertainty planning in dynamic environment. In *Proc. Int. Symp. on Robotics Research*, 2013.

[88] R. Kmmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613, May 2011.

[89] A.M. Ladd and L.E. Kavraki. Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics and Automation*, 20(2):229–242, April 2004.

[90] John J Leonard, Richard J Rikoski, Paul M Newman, and Michael Bosse. Mapping partially observable features from multiple uncertain vantage points. *The International Journal of Robotics Research*, 21(10-11):943–975, 2002.

[91] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[92] Mingyang Li and Anastasios I. Mourikis. High-precision, consistent ekf-based visualinertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[93] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Auton. Robots*, 4(4):333–349, October 1997.

[94] I. Mahon, S. B. Williams, O. Pizarro, and M. Johnson-Roberson. Efficient view-based slam using visual loop closures. *IEEE Transactions on Robotics*, 24(5):1002–1014, Oct 2008.

[95] Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186, 2007.

[96] Donald D. Marquardt. An algorithm for least squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[97] Agostino Martinelli, Viet Nguyen, Nicola Tomatis, and Roland Siegwart. A relative map approach to slam based on shift and rotation invariants. *Robotics*

and Autonomous Systems, 55(1):50–61, 2007.

[98] Jorge L Martínez, Jesús Morales, Anthony Mandow, and Alfonso García-Cerezo. Incremental closed-form solution to globally consistent 2d range scan mapping with two-step pose estimation. In *Advanced Motion Control, 2010 11th IEEE International Workshop on*, pages 252–257. IEEE, 2010.

[99] Ruben Martinez-Cantin, Nando Freitas, Eric Brochu, José Castellanos, and Arnaud Doucet. A bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(2):93–103, 2009.

[100] Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Eighteenth National Conference on Artificial Intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[101] Michael Montemerlo, Sebastian Thrun, Daphne Roller, and Ben Wegbreit. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1151–1156, San Francisco, CA, USA, 2003.

[102] Daniele Mortari. A fast on-board autonomous attitude determination system based on a new star-id technique for a wide fov star tracker. In *Sixth Annual AIAA/AAS Space Flight Mechanics Meeting, Austin, TX*, February 1996.

[103] Anastasios I Mourikis and Stergios I Roumeliotis. A dual-layer estimator architecture for long-term localization. In *Computer Vision and Pattern Recognition*

Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on, pages 1–8. IEEE, 2008.

[104] Raul Mur-Artal and Juan Tardos. Probabilistic semi-dense mapping from highly accurate feature-based monocular slam. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[105] J. Neira and J.D. Tardos. Data association in stochastic mapping using the joint compatibility test. *Robotics and Automation, IEEE Transactions on*, 17(6):890–897, Dec 2001.

[106] Paul M. Newman and Hugh F. Durrant-Whyte. Geometric projection filter: an efficient solution to the slam problem. *Proc. SPIE*, 4571:22–33, 2001.

[107] J. M. O'Kane and S. M. LaValle. Localization with limited sensing. *IEEE Transactions on Robotics*, 23(4):704–716, Aug 2007.

[108] E. B. Olson. Real-time correlative scan matching. In *2009 IEEE International Conference on Robotics and Automation*, pages 4387–4393, May 2009.

[109] Sylvie CW Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *The International Journal of Robotics Research*, 29(8):1053–1068, 2010.

[110] F. Pappalardi, S. J. Dunham, M. E. LeBlang, T. E. Jones, J. Bangert, and G. Kaplan. Alternatives to gps. In *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, volume 3, pages 1452–1459 vol.3, 2001.

[111] S. Patil, J. van den Berg, and R. Alterovitz. Estimating probability of collision for safe motion planning under gaussian motion and sensing uncertainty. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3238–3244, May 2012.

[112] V. Pilania and K. Gupta. A localization aware sampling strategy for motion planning under uncertainty. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 6093–6099, Sept 2015.

[113] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*, pages 1025–1032, Acapulco, Mexico, 2003.

[114] R. Platt. Convex receding horizon control in non-Gaussian belief space. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2012.

[115] R. Platt, L. Kaelbling, T. Lozano-Perez, , and R. Tedrake. Efficient planning in non-Gaussian belief spaces and its application to robot grasping. In *Proc. of International Symposium of Robotics Research, (ISRR)*, 2011.

[116] Robert Platt, Leslie Kaelbling, Tomas Lozano-Perez, and Russ Tedrake. Non-gaussian belief space planning: Correctness and complexity. In *ICRA*, 2012.

[117] Robert Platt, Russ Tedrake, Leslie Kaelbling, and Tomas Lozano-Perez. Belief space planning assuming maximum likelihood observatoins. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2010.

[118] Cdric Pradalier and Sepanta Sekhavat. Simultaneous localization and mapping using the geometric projection filter and correspondence graph matching. *Advanced Robotics*, 17(7):675–690, 2003.

[119] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11-12), October 2009.

[120] I. P. Prikhodko, S. A. Zotov, A. A. Trusov, and A. M. Shkel. What is mems gyrocompassing? comparative analysis of maytagging and carouseling. *Journal*

of Microelectromechanical Systems, 22(6):1257–1266, December 2013.

[121] Mohammadhussein Rafieisakhaei, Amirhossein Tamjidi, Suman Chakravorty, and PR Kumar. Feedback motion planning under non-gaussian uncertainty and non-convex state constraints. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4238–4244. IEEE, 2016.

[122] J. Reuter. Mobile robot self-localization using pdab. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 4, pages 3512–3518 vol.4, 2000.

[123] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for pomdps. *Journal of Artificial Intelligence Research*, pages 663–704, 2008.

[124] Stergios I Roumeliotis and George A Bekey. Bayesian estimation and kalman filtering: A unified framework for mobile robot localization. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2985–2992. IEEE, 2000.

[125] Amirhossein Tamjidi Saurav Agarwal and Suman Chakravorty. Motion planning for active data association and localization in non-gaussian belief spaces. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR) 2016, San Francisco, USA*, 2016.

[126] K. Schauwecker and A. Zell. On-board dual-stereo-vision for autonomous quadrotor navigation. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pages 333–342, May 2013.

[127] P. Sermanet, P. Sermanet, R. Hadsell, M. Scoffier, M. Scoffier, U. Muller, and Y. LeCun. Mapping and planning under uncertainty in mobile robots with

long-range perception. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2525–2530, Sept 2008.

[128] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[129] A. Sieverling, N. Kuhnen, and O. Brock. Sensor-based, task-constrained motion generation under uncertainty. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4348–4355, May 2014.

[130] Deborah A Sigel and David Wettergreen. Star tracker celestial localization system for a lunar rover. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[131] R. Sim and N. Roy. Global a-optimal robot exploration in slam. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 661–666, April 2005.

[132] Paul Smith, Ian D Reid, and Andrew J Davison. Real-time monocular slam with straight lines.

[133] Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous Robot Vehicles*, pages 167–193. Springer, 1990.

[134] Trey Smith and Reid Simmons. Heuristic search value iteration for pomdps. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 520–527. AUAI Press, 2004.

[135] C. Stachniss, D. Hahnel, and W. Burgard. Exploration with active loop-closing for fastslam. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceed-*

ings. 2004 IEEE/RSJ International Conference on, volume 2, pages 1505–1510 vol.2, Sept 2004.

[136] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[137] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[138] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y. Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716, 2004.

[139] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006.

[140] E. Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 300–306 vol. 1, June 2005.

[141] A. Trebi-Ollennu, T. Huntsberger, Yang Cheng, E. T. Baumgartner, B. Kennedy, and P. Schenker. Design and analysis of a sun sensor for planetary rover absolute heading detection. *IEEE Transactions on Robotics and Automation*, 17(6):939–947, Dec 2001.

[142] R. Valencia, J. Valls Mir, G. Dissanayake, and J. Andrade-Cetto. Active pose slam. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1885–1891, Oct 2012.

[143] Jur van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2010.

[144] Jur van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *IJRR*, 30(7):895–913, 2011.

[145] Jur van den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.

[146] Richard Volpe. Navigation results from desert field tests of the rocky 7 mars rover prototype. *The International Journal of Robotics Research*, 18(7):669–683, 1999.

[147] Zhan Wang, Shoudong Huang, and Gamini Dissanayake. D-slam: A decoupled solution to simultaneous localization and mapping. *The International Journal of Robotics Research*, 26(2):187–204, 2007.

[148] Wikipedia. Gyrocompass. `https://en.wikipedia.org/wiki/Gyrocompass`.

[149] Stephen Williams, Vadim Indelman, Michael Kaess, Richard Roberts, John J. Leonard, and Frank Dellaert. Concurrent filtering and smoothing: A parallel architecture for real-time navigation and full smoothing. *The International Journal of Robotics Research*, 2014.

[150] Y. Yang and O. Brock. Elastic roadmapsmotion generation for autonomous mobile manipulation. *Autonomous Robots*, 28:113–130, 2010.

[151] Guoxuan Zhang, Jin Han Lee, Jongwoo Lim, and Il Hong Suh. Building a 3-d line-based map using stereo slam. *IEEE Transactions on Robotics*, 31(6):1364–

1377, 2015.

[152] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

9. APPENDIX A

In this appendix, all necessary models for robot kinematics, dynamics and sensing that are relevant to the simulations and experiments are described.

9.1 Motion Models

The state propagation function, noise modeling, Jacobians etc. are presented for a variety of robot kinematic models.

9.1.1 Steered Bicycle

The steered bicycle model state is $\mathbf{x} = [x\ y\ \theta]^T$, control vector $\mathbf{u} = [V\ \delta]$ where V is forward speed and δ is the steering angle, B is the wheel base. Process noise $\mathbf{w} = [w_V\ w_\delta]^T$ where $w_V \sim \mathcal{N}(0, \sigma_V)$, and $w_\delta \sim \mathcal{N}(0, \sigma_\delta)$. The state propagation model is as follows

$$\mathbf{x}_{k+1} = \begin{bmatrix} x_k + (V + w_V)\Delta t cos(\delta + w_\delta + \theta_k) \\ y_k + (V + w_V)\Delta t sin(\delta + w_\delta + \theta_k) \\ \theta_k + (V + w_V)\Delta t sin(\delta + w_\delta)/B \end{bmatrix}. \tag{9.1}$$

The noise standard deviations are themselves defined as,

$$\begin{bmatrix} \sigma_V \\ \sigma_\delta \end{bmatrix} = \begin{bmatrix} \sigma_{b,V} + \eta_V V \\ \sigma_{b,\delta} + \eta_\delta \delta + \eta_{\delta,V} V \end{bmatrix}. \tag{9.2}$$

$\sigma_{b,V}, \sigma_{b,\delta}$ are the bias standard deviations, η_V, η_δ are the scaling factors such that higher control signals incur higher noise, and $\eta_{\delta,V}$ models the effect of higher speed on angular control. This models the effect that even when robot is driving in a straight, a higher speed will lead to higher perturbations in steering. The state transition

Jacobian is

$$A = \frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -V\Delta t sin(\delta + \theta_k) \\ 0 & 1 & V\Delta t cos(\delta + \theta_k) \\ 0 & 1 & 1 \end{bmatrix} \tag{9.3}$$

The control Jacobian is

$$B = \frac{\partial f}{\partial \mathbf{u}} = \begin{bmatrix} \Delta t cos(\delta + \theta_k) & -V\Delta t sin(\delta + \theta_k) \\ \Delta t sin(\delta + \theta_k) & V\Delta t cos(\delta + \theta_k) \\ \Delta t sin(\delta)/B & V\Delta t cos(\delta)/B \end{bmatrix} \tag{9.4}$$

The noise Jacobian is

$$G = \frac{\partial f}{\partial \mathbf{w}} = \begin{bmatrix} 1 & 0 & -V\Delta t sin(\delta + \theta_k) \\ 0 & 1 & V\Delta t cos(\delta + \theta_k) \\ 0 & 1 & 1 \end{bmatrix} \tag{9.5}$$

9.2 Observation Models

We describe observation models used in this research and derive properties for use in estimation and control.

9.2.1 2D Range Bearing

Let i-th feature position be $\mathbf{l}_i = [x_i \ y_i]^T$, the measurement $z^i = [r \ \phi]^T + \mathbf{v}$ gives range and bearing.

$$z^i = \begin{bmatrix} r \\ \phi \end{bmatrix} = \begin{bmatrix} \sqrt{(x_i - x)^2 + (y_i - y)^2} + v_r \\ tan^{-1}(\frac{y_i - y}{x_i - x}) - \theta + v_\theta \end{bmatrix} \tag{9.6}$$

The measurement robot state Jacobian is

$$H_R^i = \frac{\partial h}{\partial \mathbf{x}} = \begin{bmatrix} -(x_i - x)/r & -(y_i - y)/r & 0 \\ (y_i - y)/r^2 & -(x_i - x)/r^2 & -1 \end{bmatrix} \tag{9.7}$$

The measurement feature Jacobian is

$$H_m^i = \frac{\partial h}{\partial \mathbf{m}} = \begin{bmatrix} (x_i - x)/r & (y_i - y)/r \\ -(y_i - y)/r^2 & (x_i - x)/r^2 \end{bmatrix} \tag{9.8}$$

The measurement noise Jacobian is

$$M = \frac{\partial h}{\partial \mathbf{v}} = I_{2\times 2} \tag{9.9}$$

9.2.1.1 Inverse 2D Range Bearing

The inverse observation model g returns the location of the feature given robot pose and measurement.

$$\mathbf{l}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x + (r + v_r)cos(\theta + \phi + v_\phi) \\ y + (r + v_r)sin(\theta + \phi + v_\phi) \end{bmatrix} \tag{9.10}$$

Inverse observation robot state Jacobian

$$G = \frac{\partial g}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -rsin(\theta + \phi) \\ 0 & 1 & rcos(\theta + \phi) \end{bmatrix} \tag{9.11}$$

Inverse observation noise Jacobian

$$W = \frac{\partial g}{\partial \mathbf{v}} = \begin{bmatrix} cos(\theta + \phi) & -rsin(\theta + \phi) \\ sin(\theta + \phi) & rcos(\theta + \phi) \end{bmatrix} \tag{9.12}$$

10. APPENDIX B

10.1 Relative Measurements and Rotations in 2D

Parameterizing the Direction Cosine Matrix: Let the rotation from pose x_p to x_q be $\delta\theta$. The DCM $\mathbf{C}_{qp}$ for the relative rotation $\delta\theta$ between x_p and x_q is,

$$\mathbf{C}_{qp} = \begin{bmatrix} cos(\delta\theta) & -sin(\delta\theta) \\ sin(\delta\theta) & cos(\delta\theta) \end{bmatrix}. \tag{10.1}$$

Thus in planar scenarios the matrix $\mathbf{C}_{qp}$ is parameterized by the 2-vector $\mathbf{c}_{qp} = [cos(\delta\theta), sin(\delta\theta)]^T$.

Relative Feature Measurements-based Constraints on Orientation: Let a robot make observations to two landmarks l_i and l_j from poses x_p and x_q as shown in Fig. 6.2(b). Observing this pair of landmarks from both poses forms a relative orientation constraint $\mathbf{C}_{qp}$ between x_p and x_q. Let ${}^l\mathbf{d}_p^{ij}$ and ${}^l\mathbf{d}_q^{ij}$ be the relative feature measurements made from x_p and x_q respectively, then we have the following relation ${}^l\mathbf{d}_p^{ij} = \mathbf{C}_{qp}{}^l\mathbf{d}_q^{ij}$. Using Eq. 10.1 in this relation and rearranging, we have the following constraint on the relative orientation parameters,

$$\begin{bmatrix} {}^l d_{p,x}^{ij} \\ {}^l d_{p,y}^{ij} \end{bmatrix} = \underbrace{\begin{bmatrix} {}^l d_{q,x}^{ij} & -{}^l d_{q,y}^{ij} \\ {}^l d_{q,y}^{ij} & {}^l d_{q,x}^{ij} \end{bmatrix}}_{\mathbf{B}'_{qp}} \begin{bmatrix} cos(\delta\theta) \\ sin(\delta\theta) \end{bmatrix}. \tag{10.2}$$

Projection onto SO(2) Manifold: As discussed in Section 6.2.2.1, solving Eq. 6.5 or Eq. 6.3 does not provide an orthogonal rotation as the solution. Thus the linear least squares solution $\hat{\mathbf{c}}$ is projected back on the SO2 manifold by normalization

$$\hat{\mathbf{c}}_{normalized} = \boldsymbol{\eta}(\hat{\mathbf{c}}) = \frac{\hat{\mathbf{c}}}{||\hat{\mathbf{c}}||}. \tag{10.3}$$

Followed by computing the Jacobian

$$\bar{\nabla}\boldsymbol{\eta} = \frac{1}{\sqrt{c_1^2 + c_2^2}} \begin{bmatrix} c_2^2 & -c_2 c_1 \\ -c_2 c_1 & c_1^2 \end{bmatrix}, \tag{10.4}$$

and then transforming the covariance given by the linear problem as $\boldsymbol{\Sigma}_{\mathbf{c}_{normalized}} = \bar{\nabla}\boldsymbol{\eta}\Sigma_{\mathbf{c}}\bar{\nabla}^T\boldsymbol{\eta}$. We drop the normalized subscript for readability. From the projected DCM parameters we can compute the rotation angle

$$\delta\hat{\theta} = tan^{-1}(\frac{c_2}{c_1}). \tag{10.5}$$